BEST AVAILABLE IMAGES

Defective images within this document are accurate representations of the original documents submitted by the applicant.

Defects in the images may include (but are not limited to):

- BLACK BORDERS
- TEXT CUT OFF AT TOP, BOTTOM OR SIDES
- FADED TEXT
- ILLEGIBLE TEXT
- SKEWED/SLANTED IMAGES
- COLORED PHOTOS
- BLACK OR VERY BLACK AND WHITE DARK PHOTOS
- GRAY SCALE DOCUMENTS

IMAGES ARE BEST AVAILABLE COPY.

As rescanning documents *will not* correct images, please do not report the images to the Image Problem Mailbox.

PUB. NO.: 10-124484 [JP 10124484 A]
PUBLISHED: May 15, 1998 (19980515)
INVENTOR(s): ARAKAWA FUMIO
NAKAGAWA NORIO
YAMADA TETSUYA
TOTSUKA YONETARO
APPLICANT(s): HITACHI LTD [000510] (A Japanese Company or Corporation), JP (Japan)
APPL. NO.: 08-273432 [JP 96273432]
FILED: October 16, 1996 (19961016)
INTL CLASS: [6] G06F-017/10; G06F-007/38; G06F-007/52; G06F-009/38; G06F-009/38; G06T-001/00
JAPIO CLASS: 45.4 (INFORMATION PROCESSING -- Computer Applications); 45.1 (INFORMATION PROCESSING -- Arithmetic Sequence Units); 45.9 (INFORMATION PROCESSING -- Other)

ABSTRACT

PROBLEM TO BE SOLVED: To perform a vector conversion operation or an inner product operation using the floating points at a high speed and with high accuracy.

SOLUTION: The result of an inner product operation or a vector conversion operation is obtained by giving eight floating points to four multipliers 220a to 220d for their parallel operations and adding together these multiplication results in parallel to each other via a single 4-input adder 226. A circuit 227 normalizes and rounds the output of the adder 226. Thus, an inner product can be calculated via the multiplication and the addition which are carried out once in parallel to each other. Then no rounding process is required for the double input in every product sum operation to shorten the latency of the inner product operation and to improve the arithmetic accuracy. Furthermore, just a single circuit suffices for normalization, etc., and accordingly the increase of the circuit scale is minimized. As a result, an inner product operation, etc., using the floating points can be performed at a high speed and with high accuracy.

(19)日本国特許庁（ＪＰ）　(12)公開特許公報（Ａ）　(11)特許出願公開番号

特開平10－124484

(43)公開日　平成10年(1998)５月15日

(51)Int.Cl.[6]	識別記号	ＦＩ	
Ｇ０６Ｆ　17/10		Ｇ０６Ｆ　15/31	Ｓ
7/38		7/38	Ａ
7/52	３１０	7/52	３１０Ｃ
9/38	３１０	9/38	３１０Ｇ
	３７０		３７０Ｂ

審査請求　未請求　請求項の数13　ＯＬ　（全 21 頁）　最終頁に続く

(21)出願番号　特願平8－273432

(22)出願日　平成８年(1996)10月16日

(71)出願人　000005108
株式会社日立製作所
東京都千代田区神田駿河台四丁目６番地

(72)発明者　荒川　文男
東京都国分寺市東恋ケ窪一丁目280番地
株式会社日立製作所中央研究所内

(72)発明者　中川　典夫
東京都小平市上水本町五丁目20番１号　株式会社日立製作所半導体事業部内

(72)発明者　山田　哲也
東京都国分寺市東恋ケ窪一丁目280番地
株式会社日立製作所中央研究所内

(74)代理人　弁理士　玉村　静世

最終頁に続く

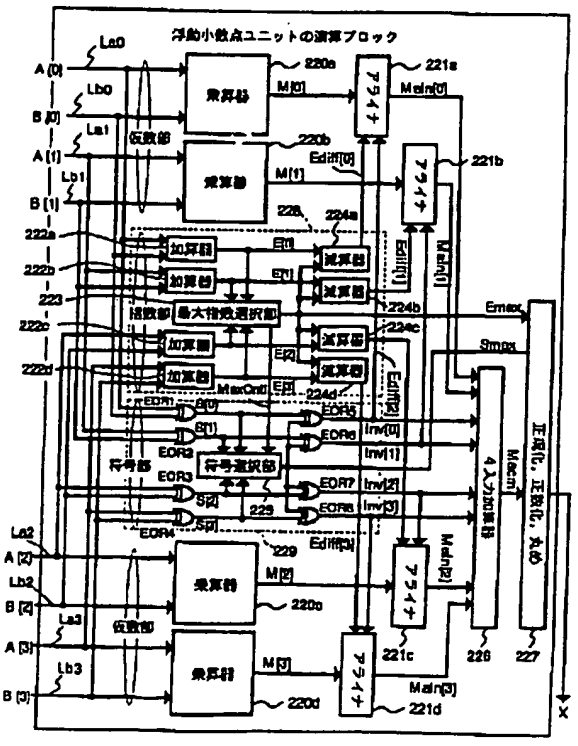
(54)【発明の名称】　データプロセッサ及びデータ処理システム

(57)【要約】

【課題】　浮動小数点数を用いたベクトル変換演算や内積演算を高速化及び高精度化する。

【解決手段】　内積演算やベクトル変換演算を行うとき、4個の乗算器（２２０ａ～２２０ｄ）に８個の浮動小数点数を与えて並列動作させ、それによる乗算結果を１個の４入力加算器２２６で並列的に加算して、演算結果を得る。４入力加算器２２６の出力に対する正規化や丸めは１個の回路（２２７）で行う。1回の並列的な乗算及び加算によって内積を求めることができる。２入力に対する積和演算毎に丸めを行う処理も必要ないから、内積演算のレイテンシーが短く、演算精度も高く、正規化等のための回路も１個備えればよいから、回路規模の増大を極力抑えて、浮動小数点による内積演算等を高速に且つ高精度に実行できる。

【図４】



【特許請求の範囲】

【請求項１】　夫々異なるデータ入力信号線群から浮動小数点数の仮数部が供給され、供給された仮数部同士の乗算を行う複数の乗算器と、夫々の乗算器の出力を受けてアライメントシフトを行うアライナと、前記アライナのアライメントシフト数及び正規化前の指数を前記浮動小数点数の指数部に基づいて生成する指数処理部と、前記アライナの出力を並列的に加算する多入力加算器と、前記多入力加算器の出力を前記正規化前の指数に基づいて正規化する正規化器とを含む演算部を浮動小数点ユニットに備えて成るものであることを特徴とするデータプロセッサ。

【請求項２】　前記演算部は更に、夫々の乗算器で乗算される浮動小数点数の符号に応じて、各乗算器の乗算結果に対する符号を生成する符号処理部を含み、前記アライナはアライメントシフト結果を選択的に反転又は非反転で出力するセレクタを有し対応する前記乗算結果に対する符合が負の場合には反転出力を選択し、前記多入力加算器は前記乗算結果に対する符合が負に対応されるアライナの出力に＋１を行うキャリーを生成して、負の乗算結果に対し２の補数化処理を行うものであることを特徴とする請求項1記載のデータプロセッサ。

【請求項３】　前記浮動小数点ユニットは更に、前記夫々の乗算器のデータ入力信号線群に接続するリードポートと前記演算部の出力に接続するライトポートとを備えるレジスタファイルを有し、このレジスタファイルは前記リードポートに並列的に接続可能な複数のバンクを有して成るものであることを特徴とする請求項１又は２記載のデータプロセッサ。

【請求項４】　前記浮動小数点ユニットは更に、夫々複数個の成分によって表されるデータ同士の内積演算を規定する浮動小数点命令を解読可能な制御部を有し、この制御部は、前記浮動小数点命令を解読して、レジスタファイルが保有するデータの成分を前記信号線群を介して前記演算部に与え、与えられたデータの内積を前記演算部に演算させ、内積の演算結果を前記レジスタファイルに書き込みさせるものであることを特徴とする請求項3記載のデータプロセッサ。

【請求項５】　前記浮動小数点ユニットは更に、夫々複数個の成分によって表されるデータと変換行列との行列演算を規定する浮動小数点命令を解読可能な制御部を有し、この制御部は、前記浮動小数点命令を解読して、レジスタファイルが保有するデータの成分と前記変換行列の成分とを読出して前記信号線群を介し前記演算部に与え、与えられたデータの内積を前記演算部に演算させ、この内積演算の結果を前記レジスタファイルに書き込みさせる一連の演算サイクルを、連続的に複数回繰返し実行させ、連続的に複数回実行される最後の演算サイクルにおける前記レジスタファイルの読出し動作が、最初の演算サイクルにおける内積演算結果を前記レジスタファイルに書き込むタイミングよりも早くなるように、前記夫々の演算サイクルのレイテンシーを制御するものであることを特徴とする請求項３記載のデータプロセッサ。

【請求項６】　前記制御部は、前記各演算サイクルにおいて、前記レジスタファイルからの読み出しを双方のバンクに対して並列的に行い、前記レジスタファイルへの書込みを一方のバンクに対して行うためのレジスタ選択制御を行うものであることを特徴とする請求項５記載のデータプロセッサ。

【請求項７】　前記レジスタファイルは各バンクに１６個のレジスタを有し、前記乗算器は４個設けられ、前記浮動小数点命令は１６ビット固定長命令であることを特徴とする請求項４乃至６の何れか１項記載のデータプロセッサ。

【請求項８】　前記浮動小数点ユニットは更に、角度データに対する正弦及び余弦をテーラ展開による多項近似に従って取得するための浮動小数点命令を解読可能な制御部を有し、この制御部は、レジスタファイルに対するレジスタリードによって角度データを演算ブロックに与え、角度データに対する正弦及び余弦を前記多項近似に従って演算ブロックに演算させ、演算結果をレジスタファイルにライトするものであり、前記角度データは、固定小数点数の小数点以下ｎビットによって１回転を２のｎ乗分割して定義するフォーマットを有し、前記多項近似は、前記角度データの小数点以下ｎビットを中心値とこの中心値に対する差分値に分けて前記角度データに応ずる正弦と余弦の値を演算するものであり、前記多項近似に必要とされる前記中心値に対する正弦又は余弦の値を保有するテーブルを更に備えて成るものであることを特徴とする請求項１乃至３の何れか１項記載のデータプロセッサ。

【請求項９】　前記小数点以下のｎビットにおける上位２ビットは角度の象限を指示し、前記制御部は、前記上位２ビットのデコード結果に従って、前記多項近似による演算結果の符号反転とその演算結果を格納するレジスタファイルのレジスタの選択を制御して、前記象限に応ずる正弦及び余弦の値を夫々に割り当てられたレジスタに格納するものであることを特徴とする請求項６記載のデータプロセッサ。

【請求項１０】　成分が夫々浮動小数点数で与えられる４×４の内積演算を１個の浮動小数点命令で実行可能な浮動小数点ユニットを含み、この浮動小数点ユニットは演算ブロックと、この演算ブロックに演算対象データを供給し且つ演算ブロックで演算された演算結果データが供給されるレジスタファイルとを含み、前記演算ブロックは、浮動小数点数の仮数部の乗算を行う４個の乗算器と、夫々の乗算器の出力を受けてアライメントシフトを行うアライナと、前記アライナのアライメントシフト数及び正規化前の指数を前記浮動小数点数の指数部に基づいて生成する指数処理部と、前記アライナの出力を並列

的に加算する４入力加算器と、前記４入力加算器の出力を前記正規化前の指数に基づいて正規化する正規化器とを含んで成るものであることを特徴とするデータプロセッサ。

【請求項１１】　成分が夫々浮動小数点数で与えられる４×４の変換行列と夫々浮動小数点数で与えられる４元の成分を持つベクトルとの積の演算を４×４の内積演算を連続４回繰り返して実行する演算を１個の浮動小数点命令で実行可能な浮動小数点ユニットを含み、この浮動小数点ユニットは演算ブロックと、この演算ブロックに演算対象データを供給し且つ演算ブロックで演算された演算結果データが供給されるレジスタファイルとを含み、前記演算ブロックは、浮動小数点数の仮数部の乗算を行う４個の乗算器と、夫々の乗算器の出力を受けてアライメントシフトを行うアライナと、前記アライナのアライメントシフト数及び正規化前の指数を前記浮動小数点数の指数部に基づいて生成する指数処理部と、前記アライナの出力を並列的に加算する４入力加算器と、前記４入力加算器の出力を前記正規化前の指数に基づいて正規化する正規化器とを含み、前記レジスタファイルは夫々16個のレジスタを含む2個のレジスタバンクを有し、前記変換行列は一方のバンクに割り当てられ、前記ベクトルは他方のバンクに割り当てられるものであることを特徴とするデータプロセッサ。

【請求項１２】　アドレスバス及びデータバスに結合されたＣＰＵを更に含み、前記浮動小数点ユニットが前記データバスに結合され、前記浮動小数点ユニットは１６ビット固定長浮動小数点命令セットを用いて浮動小数点処理を実行し、前記ＣＰＵは前記浮動小数点ユニットが浮動小数点処理を実行するための命令とデータを得るのに必要なアドレシング処理を行うものであり、１個の半導体基板に形成されて成るものであることを特徴とする請求項１乃至１１の何れか１項記載のデータプロセッサ。

【請求項１３】　請求項１２記載のデータプロセッサと、このデータプロセッサに結合され前記ＣＰＵによってアクセスされるデータＲＡＭと、前記データＲＡＭ及びデータプロセッサに結合された入出力回路とを含んで成るものであることを特徴とするデータ処理システム。

【発明の詳細な説明】

【０００１】

【発明の属する技術分野】本発明は、内積演算や行列演算に特化したデータプロセッサ、更には３次元グラフィックス制御に最適なデータ処理システムに関し、例えば４元以下の浮動小数点ベクトル又は行列を多用するアプリケーションを実行するデータプロセッサに適用して有効な技術に関するものである。

【０００２】

【従来の技術】３次元グラフィックス等では、図形の回転、拡大、縮小、透視投影及び平行移動などに４×４の変換行列を用いた行列演算を多用し、また、受光面の明るさ等を決定するのに内積演算を利用することができる。そのような行列演算や内積演算には積和演算の繰返しが必要になる。また、３次元グラフィックスで取り扱うデータについては、ハイエンドのシステムでは従来から浮動小数点数が用いられていた。ゲーム機や携帯情報端末等のようなコストの制約が厳しい分野でも、扱うデータは整数から浮動小数点数に移行しつつある。浮動小数点数を用いる方がプログラミングが容易で、高度な処理に向いているからである。

【０００３】積和演算器は、単一機能操作として（Ａ×Ｂ）＋Ｃの演算を行なうものであり、例えば、Microprocessor Report, Vol.8, No.15, November 14,1994, PP, 6-9PA-8000 Combines Complexity and Speed には積和演算ユニットを備えたプロセッサが示されているが、積和演算ユニットの並列度は２である。

【０００４】また、日経エレクトロニクス（日経ＰＢ社）,1996.1.15(No.653),pp16-17 には３次元描画機能を１チップに集積した半導体集積回路について記載がある。これには、８個の固定小数点データの演算を１サイクルで実行する積和演算器を組み込んであり、また、４×４の行列を使った座標変換を２サイクルで処理できる、と記載されている。

【０００５】

【発明が解決しようとする課題】しかしながら、上記従来技術では、浮動小数点数を用いた４×４の行列演算や内積演算などを高速化することについては考慮されていない。本発明者は、浮動小数点数を用いた行列演算や内積演算を高速化することについて検討した。それによれば、浮動小数点数の積和演算器は回路規模が大きいため、単に並列化した場合には、その回路規模の増大が著しく、上記第１の文献にも記載されるように並列度は２程度であって、高速化には限界のあることが明らかにされた。また、第２の文献に記載の内容では４×４の行列を使った座標変換を２サイクルで処理でき、ある程度の高速化は実現されているが、ビット数の少ない整数積和演算器を用いる性質上、演算精度は犠牲にならざるを得ないことが明らかにされた。

【０００６】本発明の目的は、浮動小数点数を用いた行列演算や内積演算を高速化することができるデータプロセッサを提供することにある。

【０００７】本発明の別の目的は、回路規模の増大を極力抑えて行列演算や内積演算を高精度且つ高速に処理できるデータプロセッサを提供することにある。

【０００８】本発明の前記並びにその他の目的と新規な特徴は本明細書の記述及び添付図面から明らかになるであろう。

【０００９】

【課題を解決するための手段】本願において開示される発明のうち代表的なものの概要を簡単に説明すれば下記

の通りである。

【００１０】すなわち、データプロセッサは、夫々異なるデータ入力信号線群から浮動小数点数の仮数部が供給され、供給された仮数部同士の乗算を行う複数の乗算器と、夫々の乗算器の出力を受けてアライメントシフトを行うアライナと、前記アライナのアライメントシフト数及び正規化前の指数を前記浮動小数点数の指数部に基づいて生成する指数処理部と、前記アライナの出力を並列的に加算する多入力加算器と、前記多入力加算器の出力を前記正規化前の指数に基づいて正規化する正規化器とを含む演算部を浮動小数点ユニットに備えて成るものである。

【００１１】複数の乗算器による乗算、各乗算器による乗算結果の加算が並列化されることにより、データプロセッサは、浮動小数点による内積演算やベクトル変換演算を高速化できる。また、1回の並列的な乗算及び加算によって内積を求めることができるから、２入力に対する積和演算毎に丸めを行うような処理も必要ないから、内積演算のレイテンシーが短く、演算精度も高く、また、2入力に対する積和演算毎を繰り返す場合のように演算順序が異なると演算結果も相違するという事態も生じない。しかも、データプロセッサは、正規化等のための回路を１個備えればよいから、回路規模の増大を極力抑えて、浮動小数点で内積演算やベクトル変換演算を高速に且つ高精度に行うことが可能になる。

【００１２】浮動小数点数の並列的な乗算及び加算における負数に対する処理を能率的に行う様にするには、前記演算部は更に、夫々の乗算器で乗算される浮動小数点数の符号に応じて、各乗算器の乗算結果に対する符号を生成する符号処理部を含み、前記アライナはアライメントシフト結果を選択的に反転又は非反転で出力するセレクタを有し対応する前記乗算結果に対する符合が負の場合には反転出力を選択し、前記多入力加算器は前記乗算結果に対する符合が負に対応されるアライナの出力に＋１を行うキャリーを生成して、負の乗算結果に対し２の補数化処理を行うようにするとよい。

【００１３】演算部による演算対象データ及び演算結果データはレジスタファイルに一時的に格納する。このとき、乗算器による並列乗算のためには必要なデータ全てがレジスタファイルから複数個の乗算器等に並列的に供給されなければならない。このとき、レジスタのポート数およびレジスタ指定フィールドのビット数を増やさずにそのような処理を可能にするために、レジスタファイルをレジスタバンク構成とし、複数のレジスタバンク若しくは単数のバンクのレジスタを、前記乗算器の夫々の入力端子に並列的に接続する様にすればよい。

【００１４】内積演算に着目した場合、4元以下の内積を直接求める内積演算命令をデータプロセッサの命令セットに含めるとよい。前記浮動小数点ユニットは、夫々複数個の成分によって表されるデータ同士の内積演算を規定する前記内積演算命令を解読可能な制御部を有し、この制御部は、前記浮動小数点命令を解読して、レジスタファイルが保有するデータの成分を前記信号線群を介して前記演算部に与え、与えられたデータの内積を前記演算部に演算させ、内積の演算結果を前記レジスタファイルに書き込みさせる。

【００１５】行列変換演算に着目した場合、変換行列とベクトルとの積を求めるベクトル変換演算命令をデータプロセッサの命令セットに含めるとよい。前記浮動小数点ユニットは、夫々複数個の成分によって表されるデータと変換行列との行列演算を規定するベクトル変換演算命令を解読可能な制御部を有し、この制御部は、前記浮動小数点命令を解読して、レジスタファイルが保有するデータの成分と前記変換行列の成分とを読出して前記信号線群を介し前記演算部に与え、与えられたデータの内積を前記演算部に演算させ、この内積演算の結果を前記レジスタファイルに書き込みさせる一連の演算サイクルを、連続的に複数回繰返し実行させる。このとき、ソースレジスタとディスティネーションレジスタが重なっても正しく動作できる様にするには、連続的に複数回実行される最後の演算サイクルにおける前記レジスタファイルの読出し動作が、最初の演算サイクルにおける内積演算結果を前記レジスタファイルに書き込むタイミングよりも早くなるように、前記夫々の演算サイクルのレイテンシーを制御すればよい。また、このとき、前記各演算サイクルにおいて、前記レジスタファイルからの読み出しを双方のバンクに対して並列的に行い、前記レジスタファイルへの書込みを一方のバンクに対して行うようにすれば、浮動小数点レジスタの数の不足を補うことができる。即ち、複数のオペランド（変換行列、ベクトルデータ）を複数バンクに別々に割り当ててレジスタファイルを利用する。

【００１６】そのようなレジスタファイルの利用は、１６ビット固定長浮動小数点命令のようにレジスタ指定フィールドが限られ、それ故にレジスタの数が制限されるようなアーキテクチャーに対して有用であり、且つ、そのようなリソースの制限されたアーキテクチャーのデータプロセッサにおいて浮動小数点による内積演算やベクトル変換演算を可能にしている。

【００１７】また、前記演算部に係数テーブルや乗算器のフィードバック回路を追加することにより、三角関数の近似値を区間分割と高次の多項式展開で求められるようにできる。これによれば、変換行列等に利用される正弦及び余弦を、チップ面積を大幅に増大させることなく得ることができる。

【００１８】データプロセッサは、アドレスバス及びデータバスに結合されたＣＰＵを更に含み、前記浮動小数点ユニットが前記データバスに結合され、前記浮動小数点ユニットは１６ビット固定長浮動小数点命令セットを用いて浮動小数点処理を実行する。前記ＣＰＵは前記浮

動小数点ユニットが浮動小数点処理を実行するための命令とデータを得るのに必要なアドレシング処理を行う。これにより、浮動小数点ユニットはＣＰＵと同じような高機能なアドレシングモードをサポートすることを必要とせず、この点においても、浮動小数点命令の１６ビット固定長を可能にしている。

【００１９】

【発明の実施の形態】

〔データプロセッサの構成〕図１には本発明の一例に係るデータプロセッサのブロック図が示される。同図に示されるデータプロセッサ１は、３２ビットＲＩＳＣ(Reduced Instruction Set Computer)アーキテクチャを有し、１６ビット固定長浮動小数点命令を実行する。この実施の態様は特に３次元グラフィックスを十分にサポートする必要を持つような機器組み込み制御（例えばビデオ・ゲーム）への応用に有効である。

【００２０】このデータプロセッサ１は、浮動小数点ユニット２を有する。浮動小数点ユニット２が演算を行う浮動小数点数は単精度とされる。さらに、データプロセッサ１は中央処理装置（ＣＰＵ）３を有し、このＣＰＵ３は整数を処理する能力を持つ整数ユニットとされる。前記ＣＰＵ３は３２ビットデータバス４を介して前記浮動小数点ユニット２に結合されている。ＣＰＵ３及び浮動小数点ユニット２は命令バス５を介して命令キャッシュユニット６から命令を取り込む。命令アドレスはＣＰＵ３から命令キャッシュユニット６に与えられる。データキャッシュユニット７は、前記データバス４に接続され、データアドレスバス８を介してＣＰＵ３からデータアドレスが供給される。前記データキャッシュユニット７及び命令キャッシュユニット６は、夫々図示を省略するキャッシュコントローラ及びキャッシュメモリを備えている。前記命令キャッシュユニット６及びデータキャッシュユニット７はデータ信号やコントロール信号を含むキャッシュバス１３を介してバスコントローラ９に接続される。命令キャッシュユニット６におけるキャッシュミス等に起因する外部アクセスのための命令アドレスは前記バスコントローラ９に与えられる。また、データキャッシュユニット７におけるキャッシュミス等に起因する外部アクセスのためのデータアドレスは前記バスコントローラ９に与えられる。バスコントローラ９はそれら命令アドレス又はデータアドレスに従って、代表的に図示されたアドレスピン及びデータピン等に結合される外部メモリなどをアクセスするために外部バスサイクルを起動する。また、バスコントローラ９にはタイマやシリアルコミュニケーションインタフェースコントローラ等の周辺回路１０が周辺バス１１を介して接続されている。図１に示されるデータプロセッサは、単結晶シリコンのような１個の半導体基板に形成されている。

【００２１】前記浮動小数点ユニット（以下単にＦＰＵとも称する）２は浮動小数点処理のためにメモリからデータ又は命令を要求することになる。この実施の態様において、前記ＦＰＵ２は、データキャッシュユニット７のキャッシュメモリにデータをストアし又は当該キャッシュメモリからデータを獲得するためのメモリアドレシング能力を持っていない。これは、ＦＰＵ２のメモリ・アドレシング回路の必要性を取り除くことによってチップ面積を節約するためである。それに代えて、ＣＰＵ３はＦＰＵ２に代わってキャッシュメモリなどをアドレシングする機能を有する。したがって、ＦＰＵ２若しくは浮動小数点命令は、ＣＰＵ３と同様の強力なアドレシングモードをサポートする必要はなく、その機能を全てＣＰＵ３が負担する。ＣＰＵ３はＦＰＵ２のためにメモリからデータのフェッチを行うだけでなく、ＣＰＵ３はまた、ＦＰＵ２のために浮動小数点命令を含む全ての命令をメモリからフェッチする。命令はＣＰＵ３とＦＰＵ２の双方に取り込まれてデコードされる。ＣＰＵ３は、デコードした命令がＣＰＵ命令である場合にはそれによって指示される整数処理を実行する。また、ＣＰＵ３は、デコードした命令がＦＰＵ命令である場合には、ＦＰＵ２に代わって実行すべきアドレシング処理などを行う。ＦＰＵ２は、デコードした命令がＦＰＵ命令である場合にはそれによって指示される浮動小数点処理を実行する。また、ＦＰＵ２は、デコードした命令がＣＰＵ命令である場合にはその命令を無視する。

【００２２】図２には前記データプロセッサの主なレジスタ構成が示される。ＣＰＵレジスタは１６本の汎用レジスタｒ０～ｒ１５と、それに対するカーネルバンクレジスタｋ０～ｋ７を有する。例えばカーネルバンクレジスタｋ０～ｋ７は、例外発生時に、汎用レジスタｒ０～ｒ７の退避に代え、バンク切換え制御によって利用される。

【００２３】ＦＰＵレジスタはフロントバンクとバックバンクを有するバンクレジスタ構成とされる。フロントバンクは１６本のレジスタｆ０～ｆ１５を有する。バックバンクはそれに対応する１６本のレジスタｂ０～ｂ１５を有する。何れのバンクのレジスタを利用するかは、例えばコントロールレジスタの制御ビットの値によって決定される。ＦＰＵレジスタの場合には更に、特定の浮動小数点命令を実行するとき双方のバンクのレジスタをソースレジスタ及びディスティネーションレジスタとして利用する。その詳細については後述する。フロントバンクのレジスタｆ０～ｆ１５は、単精度フォーマットの浮動小数点数に対しては１６本のレジスタとされ、倍精度フォーマットの浮動小数点数に対しては８本のレジスタ（ｄ０，ｄ２，ｄ４，ｄ８，ｄ１０，ｄ１２，ｄ１４）とされ、各成分が単精度フォーマットで与えられる４成分を持つベクトルデータに対しては４本のベクトルレジスタ（Ｖ０，Ｖ４，Ｖ８，Ｖ１２）とされる。

【００２４】また、ＦＰＵ２とＣＰＵ３によって共有されるコミュニケーションレジスタＦＰＵＣを有する。こ

のレジスタＦＰＵＣは、ＣＰＵ３とＦＰＵ２との間でのデータの受け渡しを高速化するために設けられている。尚、前記各レジスタは３２ビット構成である。

【００２５】〔ＦＰＵの構成〕図３には前記ＦＰＵ２のブロックダイヤグラムが示される。ＦＰＵ２は、転送ブロック２０、レジスタファイル２１、演算ブロック２２及び制御部２３によって構成される。演算ブロック２２はその詳細を後述するように行列演算やベクトル演算の高速化を実現した積和演算回路の構成を有する。レジスタファイル２１は図２で説明したＦＰＵレジスタを含み、演算ブロック２２に対しては８個のリードポートA[0]，B[0]，A[1]，B[1]，A[2]，B[2]，A[3]，B[3]を有し、演算ブロック２２からデータを受けるライトポートＸを有する。転送ブロック２０はレジスタファイル２１のリードポートＣから出力されるデータをデータバス４に供給するバスドライバ２００、データバス４からの入力又はレジスタファイル２１からの出力を選択してレジスタファイル２１のライトポートＹに供給するセレクタ２０１を有する。制御部２３は、命令バス５から供給された命令をデコードし、そのデコード結果に従って転送ブロック２０、レジスタファイル２１及び演算ブロック２２を制御する。バスドライバ２００及びセレクタ２０１の制御信号BusDrv及びLoadCntlも制御部２３で形成される。

【００２６】図４には演算ブロック２２の一例が示される。演算ブロック２２は、レジスタファイルのリードポートA[0]，B[0]，A[1]，B[1]，A[2]，B[2]，A[3]，B[3]（それらリードポートを単にA[n]，B[n]とも記す）に夫々個別に結合する信号線群Ｌａ０，Ｌｂ０，Ｌａ１，Ｌｂ１，Ｌａ２，Ｌｂ２，Ｌａ３，Ｌｂ３（それら信号線群を単にＬａｉ，Ｌｂｉとも記す）を有する。４個の乗算器２２０ａ～２２０ｄには夫々の信号線群Ｌａｉ，Ｌｂｉを介して、浮動小数点数の仮数が、乗数及び被乗数として供給される。乗算器２２０ａ～２２０ｄは、夫々に供給された乗数及び被乗数を乗算して、その積M[0]，M[1]，M[2]，M[3]（以下単にＭ［ｎ］とも記す）を出力する。積M[0]，M[1]，M[2]，M[3]は、夫々に対応されるアライナ２２１ａ～２２１ｄに供給される。

【００２７】夫々の信号線群Ｌａｉ，Ｌｂｉに供給される浮動小数点数の指数部は夫々加算器２２２ａ～２２２ｄに供給される。また、夫々の信号線群Ｌａｉ，Ｌｂｉに供給される浮動小数点数の符号部は夫々排他的論理和ゲートＥＯＲ１～ＥＯＲ４に供給される。

【００２８】前記加算器２２２ａ～２２２ｄ、最大指数選択部２２３及び減算器２２４ａ～２２４ｄは指数処理部２２８を構成する。加算器２２２ａ～２２２ｄは乗数と被乗数に対応される浮動小数点数の指数部を加算してその和E[0]，E[1]，E[2]，E[3]を出力する。最大指数選択部２２３は、前記指数の和E[0]，E[1]，E[2]，E[3]の内から最大のものをＥｍａｘとして選択する。減算器２２４ａ～２２４ｄはＥｍａｘからE[0]，E[1]，E[2]，E[3]を減算して差分Ｅｄｉｆｆ[0]，Ｅｄｉｆｆ[1]，Ｅｄｉｆｆ[2]，Ｅｄｉｆｆ[3]（以下単にＥｄｉｆｆ［ｎ］とも記す）を得る。前記差分Ｅｄｉｆｆ[0]，Ｅｄｉｆｆ[1]，Ｅｄｉｆｆ[2]，Ｅｄｉｆｆ[3]は、前記アライナ２２１ａ～２２１ｄによるアライメントシフト数を制御する。したがって、各アライナ２２１ａ～２２１ｄの出力Ｍａｌｎ［０］，Ｍａｌｎ［１］，Ｍａｌｎ［２］，Ｍａｌｎ［３］（以下単にＭａｌｎ［ｎ］とも記す）は、最大指数Ｅｍａｘに応じた桁位置を持つことになる。このように、指数部２２８は、前記差分Ｅｄｉｆｆ[0]，Ｅｄｉｆｆ[1]，Ｅｄｉｆｆ[2]，Ｅｄｉｆｆ[3]によって前記アライナ２２１ａ～２２１ｄのアライメントシフト数を決定すると共に、正規化前の指数Ｅｍａｘを浮動小数点数の指数部に基づいて生成する。

【００２９】前記排他的論理和ゲートＥＯＲ１～ＥＯＲ４、排他的論理和ゲートＥＯＲ５～ＥＯＲ８及び符合選択部２２５は符号処理部２２９を構成する。前記排他的論理和ゲートＥＯＲ１～ＥＯＲ４は乗数と被乗数に対応される浮動小数点数の符号部を入力して、乗数と被乗数の積の符号を判定する。判定された符号Ｓ［０］，Ｓ［１］，Ｓ［２］，Ｓ［３］は符合選択部２２５にて前記Ｅｍａｘに応ずる一つがＳｍａｘとして選択される。前記符号Ｓ［０］，Ｓ［１］，Ｓ［２］，Ｓ［３］は代表符号Ｓｍａｘとの一致が排他的論理和ゲートＥＯＲ５～ＥＯＲ８によって判定される。その判定結果Ｉｎｖ［０］，Ｉｎｖ［１］，Ｉｎｖ［２］，Ｉｎｖ［３］（以下単にＩｎｖ［ｎ］とも記す）は対応するアライナ２２１ａ～２２１ｄに供給され、例えば判定結果Ｉｎｖ［０］，Ｉｎｖ［１］，Ｉｎｖ［２］，Ｉｎｖ［３］が論理値“１”の場合にはアライナ２２１ａ～２２１ｄは、対応する積Ｍ［ｎ］を反転して出力Ｍａｌｎ［ｎ］を形成する。これは積Ｍ［ｎ］を２の補数に変換するための前処理とされる。このように、符号処理部２２９は、夫々の乗算器２２０ａ～２２０ｄで乗算される浮動小数点数の符号に応じて、正規化前の符号Ｓｍａｘ及びこの符号Ｓｍａｘに対する各乗算器の乗算結果に対する符号Ｉｎｖ［ｎ］を生成する。

【００３０】４入力加算器２２６は、前記アライナ２２１ａ～２２１ｄの出力Ｍａｌｎ［ｎ］を並列的に入力して加算する。４入力の並列加算処理に際して、前記符号Ｉｎｖ［ｎ］が供給される。詳細は後述するが、４入力加算器２２６は、前記２の補数化の前処理が行われている出力Ｍａｌｎ［ｎ］に対してその最下位に＋１するための処理を前記符号Ｉｎｖ［ｎ］に基づいて行う。

【００３１】４入力加算器２２６の出力Ｍａｃｍは、正規化、正数化及び丸め処理回路２２７に供給される。この回路２２７は、前記正規化前の指数Ｅｍａｘと加算出

力Ｍａｃｍと符号Ｓｍａｘとに基づいて正規化および正数化を行い、単精度浮動小数点フォーマットに適合する丸めを行って、浮動小数点数を得る。これによって得られる浮動小数点数は、Ａ[０]・Ｂ[０]＋Ａ[１]・Ｂ[１]＋Ａ[２]・Ｂ[２]＋Ａ[３]・Ｂ[３]の積和演算結果とされる。

【００３２】図５には最大指数選択部２２３の一例が示される。前記Ｅ［１］とＥ［０］が大小比較器２２３０によって比較され、大きい方がセレクタ２２３１で選択される。同様に、Ｅ［３］とＥ［２］が大小比較器２２３２によって比較され、大きい方がセレクタ２２３３で選択される。双方のセレクタで選択されたものは、更に大小比較器２２３４で比較され、大きい方がセレクタ２２３５で選択される。セレクタ２２３５の出力が前記正規化前の指数Ｅｍａｘとされる。

【００３３】図６には符合選択部２２５の一例が示される。セレクタ２２５０は前記Ｓ［１］又はＳ［０］を選択し、セレクタ２２５１は前記Ｓ［３］又はＳ［２］を選択し、セレクタ２２５２はセレクタ２２５０の出力又はセレクタ２２５１の出力を選択する。セレクタ２２５０～２２５２の選択制御信号は前記大小比較器２２３０，２２３２，２２３４の比較判定結果信号ＭａｘＣｎｔｌとされ、これによって、Ｅｍａｘとして選択された指数に係る浮動小数点数の符号部が、前記正規化前の符号Ｓｍａｘとして選択される。

【００３４】図７にはアライナ２２１ａ（２２１ｂ～２２１ｄ）の一例が示される。シフタ２２１０はＭ［ｎ］を入力し、Ｅｄｉｆｆ［ｎ］によってアライメントシフト数（シフトビット数）が制御される。シフタ２２１０の出力はインバータ２２１１で反転され、インバータ２２１１の出力又はシフタ２２１０の出力がＩｎｖ［ｎ］によってセレクタ２２１２で選択され、選択された値がＭａｌｎ［ｎ］とされる。

【００３５】図８には４入力加算器２２６の一例が示される。この４入力加算器２２６は、桁上げ抜きの和（和出力）と桁上げ（キャリー出力）とをキャリー保存加算器アレイ２２６０で別々に求め、キャリー伝播加算器２２６１で最終の和を得る時点まで桁上げの伝播を遅延させる回路形式を有する。この４入力加算器２２６によって得られる和Ｍａｃｍは入力のビット数に対して最大２ビット増える場合があるから、４入力加算器２２６に入力される積Ｍａｌｎ［ｎ］は予じめ２ビット符号拡張が施されてキャリー保存加算器アレイ２２６０に供給される。

【００３６】図８において、前記２の補数化のための後処理（＋１）は３ビットのキャリー信号Ｃｉｎ［０］，Ｃｉｎ［１］，Ｃｉｎ［２］によって行われる。前述の説明から明らかなように、符号選択部２２５はＳ［ｎ］の内のどれか一つを選択するから、Ｉｎｖ［ｎ］のうちの少なくとも一つは必ず論理値“０”にされる。従って、２の補数化の対象はＭａｉｎ［ｎ］の内の３個以下にしかならない。これを検出するのがＯＲゲート２２６２、ＯＲゲート２２６３、ＡＮＤ・ＯＲゲート２２６４でありる。図９にはＩｎｖ［ｎ］の値に対してＣｉｎ［０］，Ｃｉｎ［１］，Ｃｉｎ［２］の採り得る値が示されており、これによっても明らかなように、Ｃｉｎ［２］はＩｎｖ［２］とＩｎｖ［３］の少なくとも一方が論理値“１”のときに論理値“１”にされ、Ｃｉｎ［１］はＩｎｖ［１］とＩｎｖ［０］の少なくとも一方が論理値“１”のときに論理値“１”にされ、Ｃｉｎ［０］はＩｎｖ［１］とＩｎｖ［０］或いはＩｎｖ［２］とＩｎｖ［３］が共に論理値“１”のときに論理値“１”にされる。

【００３７】図１０には前記キャリー保存加算器アレイ２２６０とキャリー伝播加算器２２６１の詳細な論理構成の一例が示される。前記キャリー保存加算器アレイ２２６０は、特に制限されないが、複数個の４－２コンプレッサ（４－２ＣＯＭＰ）２２６５によって構成される。夫々の４－２コンプレッサ２２６５は図１１の（Ａ）に例示されるように５入力（Ｉ１～Ｉ４，Ｃｉ）と３出力（Ｓ，Ｃ，Ｃｏ）を有する。キャリー出力Ｃｏは隣の上位ビットのキャリー入力Ｃｉに接続するため、４－２コンプレッサ２２６５は４個のビットＩ１～Ｉ４を加算する。Ｓはその加算出力、Ｃはその加算によって生ずるキャリー出力である。４－２コンプレッサ２２６５の中では、ＣｏがＣｉに依存しないので、見掛け上、キャリー伝達はない。例えば１個の４－２コンプレッサ２２６５は、図１１の（Ｂ）に例示されるように、２個の全加算器によって構成することができる。全加算器は、特に制限されないが、（Ｃ）に例示されたマルチプレクサＭＵＸを３個用いて構成される。尚、４－２コンプレッサについて記載された文献としては信学技報（電子情報通信学会）TECHNICAL REPORT OF IEICE ICD94-135, DSP94-91(1994-10)の「パストランジスタ・マルチプレクサを適用した校則５４×５４ビット乗算器（第73～79頁）がある。

【００３８】図１０において、夫々の４－２コンプレッサ２２６５には、前記アライナ出力Ｍａｌｎ［０］～Ｍａｌｎ［３］における同一桁位置のビットが下位側から順次４ビット単位で供給されている。Ｍａｌｎ［０］０～Ｍａｌｎ［３］０はＭａｌｎ［０］～Ｍａｌｎ［３］における最下位の４ビットを意味している。前記キャリー信号Ｃｉｎ［３］は最下位の４－２コンプレッサ２２６５のキャリー入力端子Ｃｉに与えられる。前記キャリー伝播加算器２２６１は複数個の全加算器２２６１によって構成され、キャリー出力は上位の全加算器のキャリー入力とされる。全加算器の一方の加算入力は４－２コンプレッサ２２６５の和出力Ｓとされ、もう一方の加算入力は一つ上位に配置された４－２コンプレッサ２２６５のキャリー出力Ｃとされる。前記キャリー信号Ｃｉｎ

〔2〕は最下位の全加算器の一方の加算入力信号として与えられ、前記キャリー信号Ｃｉｎ〔1〕は最下位の全加算器のキャリー入力信号として与えられる。

【００３９】図１２にはレジスタファイルの一例ブロックダイヤグラムが示され、図１３にはレジスタファイルを構成する各レジスタグループの構成が示され、図１４にはレジスタグループの各レジスタ回路の構成が示される。

【００４０】レジスタファイル２１は、特に制限されないが、図１２に示されるように４個のレジスタグループFR-Gr.[0]～FR-Gr.[3]を有し、各レジスタグループFR-Gr.[m]は図１３に示されるように4個のレジスタ回路FR[m], FR[m+4], FR[m+8], FR[m+12]を有する。図１３においてmは０～３の整数である。夫々のレジスタ回路は図１４に示されるように、フロントバックとバックバンクを構成するための一対のレジスタFRJ[n], FRK[n]を有する。図１４においてnは０～１５の整数である。レジスタFRJ[n], FRK[n]に対する書込み動作の指示は信号Write[n]によって与えられる。書込み対象とされるレジスタは信号Bankによって何れか一方が選択される。レジスタFRJ[n], FRK[n]の出力と端子P[n], Q[n]の対応は信号BankによってセレクタＳＬ１，ＳＬ２で交互に切換え可能にされる。図１３に示されるように1個のレジスタグループにおいて、4個のレジスタ回路の端子P[m], P[m+4], P[m+8], P[m+12]は、2ビットの信号ReadAによりセレクタＳＬ３で何れか1個が選択されることによって端子R[m]に接続可能にされ、同様に2ビットの信号ReadBによりセレクタＳＬ４で何れか1個が選択されることによって端子B[m]に接続可能にされる。レジスタグループの端子Q[m], Q[m+4], Q[m+8], Q[m+12]は信号ReadAによって制御されるセレクタＳＬ５～ＳＬ８でレジスタグループ単位に選択される。前記セレクタＳＬ５，ＳＬ６，ＳＬ７，ＳＬ８の出力と夫々のレジスタグループの出力R[3], R[2], R[1], R[0]とは信号ReadTypeによって制御されるセレクタＳＬ９，ＳＬ１０，ＳＬ１１，ＳＬ１２で選択され、選択されたものがリードポートA[3], A[2], A[1], A[0]の出力とされる。したがって、リードポートA[3], A[2], A[1], A[0]からは、図１５に示されるように、レジスタグループ単位で4個の浮動小数点レジスタから並列的にデータをリードし、或いは、個々のレジスタグループから1個づつ並列的にデータをリードすることができる。また、夫々のレジスタグループの出力B[3], B[2], B[1], B[0]はそのままリードポートB[3], B[2], B[1], B[0]の出力とされる。したがって、リードポートB[3], B[2], B[1], B[0]からは、図１６に示されるように、個々のレジスタグループから1個づつ並列的にデータをリードすることができる。前記リードポートＣには夫々のレジスタグループの出力B[3], B[2], B[1], B[0]が信号ReadCによってセレクタＳＬ１３で選択されたものが接続される。したがって、図１７に示されるように、信号ReadBとReadCとの状態に応じてレジスタを任意に選択してポートＣから読み出すことができる。前記ライトポートＸ，Ｙからの入力は信号WriteTypeによって制御されるセレクタＳＬで選択される。

【００４１】〔内積演算〕前記ＦＰＵ２を用いた内積演算について説明する。例えば内積は、図１８に示されるように、3次元空間において、ある特定の面に光を当てた時の面の明るさを求めるのに利用できる。ＦＰＵ２は、ベクトルＶ１（＝〔Ｘ１，Ｙ１，Ｚ１，Ｗ１〕）とＶ２（＝〔Ｘ２，Ｙ２，Ｚ２，Ｗ２〕）との内積ｉを１個の浮動小数点内積演算命令（単に内積演算命令とも称する）ｆｔｐｒ　Ｖｎ，Ｖｍによって求めることができる。

【００４２】前記内積演算命令による処理の概略は図１９に示される。例えば、レジスタファイル２１のベクトルレジスタＶ０に〔Ｘ１，Ｙ１，Ｚ１，Ｗ１〕が、Ｖ４に〔Ｘ２，Ｙ２，Ｚ２，Ｗ２〕がロードされているものとする。前記内積演算命令が制御部２３で解読されると、レジスタファイル２１のリード動作が制御されて、乗算器２２０ａにＸ１とＸ２が、乗算器２２０ｂにＹ１とＹ２が、乗算器２２０ｃにＺ１とＺ２が、乗算器２２０ｄにＷ１とＷ２が、夫々並列的に供給される。図１９では前記指数処理部や符号処理部などの図示を省略しているが、並列的な乗算結果は前記アライナによるシフトや反転等を経て4入力加算器２２６で加算され、その加算結果に対して正規化等が行われて内積が得られる。得られた内積は、ベクトルレジスタＶ０の内、Ｗ１の値を保有するレジスタにポートＸを介して上書きされる。このように、浮動小数点の積和演算が並列的に行われるので、内積演算を高速化できる。

【００４３】図２０には前記内積演算命令におけレジスタファイルの利用に関する仕様の一例が示される。即ち、ベクトルＶ〔m〕とＶ〔n〕の内積演算結果を浮動小数点レジスタＦＲ〔ｎ＋３〕に格納する。ベクトルＶ〔n〕の成分は浮動小数点レジスタＦＲ〔ｎ〕，ＦＲ〔ｎ＋１〕，ＦＲ〔ｎ＋２〕，ＦＲ〔ｎ＋３〕にロードされる。ここでｎは０，４，８，１２の何れかであり、またＦＲ〔ｎ〕は前記フロントバンクのレジスタｆｎに対応されるものと理解されたい。レジスタファイル２１の構成上、レジスタＦＲ〔ｎ〕は、制御信号Bank＝０の場合にはレジスタＦＲＪ〔ｎ〕に割り当てられ、制御信号Bank＝１の場合にはレジスタＦＲＫ〔ｎ〕に割り当てられる。この仕様において、例えば図１５のBank＝０、ReadA＝０でポートＡからの出力が指定されたるレジスタＦＲＪ〔０〕，ＦＲＪ〔１〕，ＦＲＪ〔２〕，ＦＲＪ〔３〕にベクトルデータＶ〔n〕を置き、図１６のBank＝０、ReadB＝１でポートＢからの出力が指定されたレジスタＦＲＪ〔４〕，ＦＲＪ〔５〕，ＦＲＪ〔６〕，ＦＲＪ〔７〕にベクトルデータＶ〔m〕を置けば、Ｖ〔n〕とＶ〔m〕の内積演算に必要な８個の成分データ

を並列的に演算ブロックに与えて上述の内積演算を行うことができる。演算に際して実際にどのレジスタをリードするかは内積演算命令のレジスタ指定フィールドで指定される。そのレジスタ指定フィールドにおけるレジスタの指定には、ソースレジスタとディスティネーションレジスタの指定に４ビットを用いる。

【００４４】〔ベクトル変換演算〕次に、前記ＦＰＵ２を用いたベクトル変換演算について説明する。周知の４行４列の変換行列は並進、回転、伸張、及び透視等の変換を表す事ができ、この変換行列とベクトルの積によって、その変換行列が表すベクトル変換を得ることができる。ベクトル変換演算は一般に図２１で示されるように表すことができる。Ａは変換行列、Ｐは変換対象とされるデータ、Ｐ’は変換後のデータである。そのようなベクトル変換は１個の浮動小数点ベクトル変換演算命令（単にベクトル変換演算命令とも称する）ｆｔｒｖ　ｂａｃｋ，Ｖｎによって求めることができる。

【００４５】前記ベクトル変換演算命令による処理の概略は図２２に示される。例えば、変換行列はバックバンクの１６本のレジスタに配置される。そしてベクトルデータ［Ｘｉ，Ｙｉ，Ｚｉ，Ｗｉ］はフロントバンクを構成するレジスタに格納される。

【００４６】このベクトル変換演算命令による処理は、実質的に４回の内積演算を順次繰り返す処理に等しい。即ち、［Ｘｉ，Ｙｉ，Ｚｉ，Ｗｉ］×［ａ１１，ａ１２，ａ１３，ａ１４］を演算してその結果をＸｉの領域にライト、［Ｘｉ，Ｙｉ，Ｚｉ，Ｗｉ］×［ａ２１，ａ２２，ａ２３，ａ２４］を演算してその結果をＹｉの領域にライト、［Ｘｉ，Ｙｉ，Ｚｉ，Ｗｉ］×［ａ３１，ａ３２，ａ３３，ａ３４］を演算してその結果をＺｉの領域にライト、［Ｘｉ，Ｙｉ，Ｚｉ，Ｗｉ］×［ａ４１，ａ４２，ａ４３，ａ４４］を演算してその結果をＷｉの領域にライト、の処理を順次実行する。夫々の処理は実質的に内積演算処置と同じである。

【００４７】前記ベクトル変換演算命令が制御部２３で解読されると、上記最初の内積演算処理を行うためのデータがレジスタファイル２１から乗算器２２０ａ～２２０ｄ等に夫々並列的に供給される。図２２では同じく前記指数処理部や符号処理部などの図示を省略しているが、並列的な乗算結果は前記アライナによるシフトや反転等を経て4入力加算器２２６で加算され、その加算結果に対して正規化等が行われて内積が得られる。得られた内積は、Ｘｉを保有するレジスタにライトされる。このような処理をレジスタファイルのリード対象レジスタとライト対象レジスタを順次変更しながら繰り返す。このように、浮動小数点の内積処理を4回連続的に繰り返すことにより、ベクトル変換の結果を高速に得ることができる。

【００４８】図２３には前記ベクトル変換演算命令におけるレジスタファイルの利用に関する仕様の一例が示される。即ち、ベクトルＶ［ｎ］と変換行列Matrixとの積をレジスタＶ［ｎ］に上書きする。ベクトルＶ［ｎ］の成分は浮動小数点レジスタＦＲ［ｎ］，ＦＲ［ｎ＋１］，ＦＲ［ｎ＋２］，ＦＲ［ｎ＋３］にロードされる。ここでｎは０，４，８，１２の何れかであり、またＦＲ［ｎ］は前記フロントバンクのレジスタｆｎに対応されるものと理解されたい。変換行列はバックバンクを構成するレジスタＦＢ［０］～ＦＢ［１５］（図２のｂ０～ｂ１５に対応されるレジスタ）に格納される。レジスタファイル２１の構成上、レジスタＦＢ［ｎ］は、制御信号Bank＝０の場合にはレジスタＦＲＫ［ｎ］に割り当てられ、制御信号Bank＝１の場合にはレジスタＦＲＪ［ｎ］に割り当てられる。

【００４９】この仕様において、変換行列Matrixは、図１５においてReadType＝１の状態でポートＡから並列的に出力され、ベクトルＶ［ｎ］はポートＢから並列的に出力される。例えば最初の内積演算では、図15を参照すれば、ReadType＝１，Bank＝１，ReadA＝０によって、ＦＲＪ［０］，ＦＲＪ［４］，ＦＲＪ［８］，ＦＲＪ［１２］から変換行列Matrixの第1行目がポートＡから出力され、これに並行して、Bank＝１，ReadB＝０によって、ＦＲＫ［０］，ＦＲＫ［１］，ＦＲＫ［２］，ＦＲＫ［３］から変換対象ベクトル［Ｘｉ，Ｙｉ，Ｚｉ，Ｗｉ］がポートＢから出力される。順次これに続く3回の内積演算では、それ毎に、ReadAによる選択を１，２，３のように変化させればよい。Ｂポートからのリード対象レジスタは４回の内積演算処理において同一とされる。

【００５０】一つのベクトル変換命令による前記複数回の内積演算処理は図２４に示されるようにパイプライン処理で行われる。つまり、１つの命令で、４つのパイプライン処理が実行される。内積演算処理の一つのパイプは、レジスタリードステージＲＲ、第１演算ステージＦ１、第２演算ステージＦ２、第３演算ステージＦ３、レジスタライトステージＲＷ．及び図示を省略する命令フェッチステージとされる。命令フェッチステージは当然レジスタリードステージＲＲの前に配置され、また、レジスタリードステージＲＲは命令のデコード処理も含むことになる。この例では、乗算から正規化までの演算を３個の演算ステージを経て行うことになる。図２４の（１）のパイプラインで実行される処理は、（ＦＢ[0]，ＦＢ[４]，ＦＢ[８]，ＦＢ[１２]）×Ｖ[n]の内積演算を行ってその結果をレジスタＦＲ[ｎ]にライトし、（２）のパイプラインで実行される演算処理は、（ＦＢ[1]，ＦＢ[5]，ＦＢ[9]，ＦＢ[13]）×Ｖ[n]の内積演算を行ってその結果をＦＲ[ｎ+1]にライトし、（３）のパイプラインで実行される演算処理は、（ＦＢ[2]，ＦＢ[6]，ＦＢ[10]，ＦＢ[14]）×Ｖ[n]の内積演算を行ってその結果をＦＲ[ｎ+2]にライトし、（４）のパイプラインで実行される演算処理は、（ＦＢ[3]，ＦＢ[7]，ＦＢ

[11]，ＦＢ[15]）×Ｖ[n]の内積演算を行ってその結果をＦＲ[ｎ+3]にライトするものとされる。ディスティネーションレジスタＦＲ[n]，ＦＲ[n+1]，ＦＲ[n+2]，ＦＲ[n+3]はＶ[n]のソースレジスタでもある。

【００５１】このとき、一連の４回の内積演算処理におきて、先頭のパイプライン（１）におけるレジスタライトＲＷは、最後のパイプライン（４）におけるレジスタリードＲＲの後にされている。換言すれば、Ｖ[n]の成分と前記変換行列の成分とをレジスタファイル２１から読出して内積演算を行い当該内積演算の結果をレジスタファイルにライトする一連の演算サイクルを、連続的に複数回繰返し実行させるとき、連続的に複数回実行される最後の演算サイクルにおける前記レジスタファイルの読出し動作が、最初の演算サイクルにおける内積演算結果を前記レジスタファイルに書き込むタイミングよりも早くなるように、前記夫々の演算サイクルのレイテンシーが制御される。したがって、ソースレジスタとディスティネーションレジスタが同一レジスタであっても、ソースレジスタから全てのデータリードされるまではライトは行われず、データＶ[n]が不所望に失われることはない。

【００５２】ベクトル変換演算におけるベクトルデータのソースレジスタとディスティネーションレジスタを同一にしても、動作上支障はない。変換前後のベクトルデータを同一レジスタに配置できるので、ベクトル変換処理を多用するプログラムの作成が容易になる。また、3次元グラフィックス等におけるベクトル変換演算は多数のベクトル若しくは点に対して行われることになる。このとき、変換前後のデータが同一レジスタに配置されれば、16本のフロントバックを構成する浮動小数点レジスタに4個のベクトルデータをロードすれば、ベクトル変換命令を4回連続的に実行することができる。即ち、そのような16本の浮動小数点レジスタに対する演算対象データのロード又は演算結果データのメモリへのストア動作の回数が少なくて済む。これに対して、変換前後のデータを別のレジスタの格納する場合には、一つのベクトル変換命令の実行に８本の浮動小数点レジスタを費やす結果、演算対象データのロードや演算結果データのストア動作の頻度が多くなってしまう。この意味において、ベクトル変換演算におけるベクトルデータのソースレジスタとディスティネーションレジスタを同一にできることは、レジスタ本数が限られた中で、ベクトル変換演算を高速化するのに有用である。

【００５３】〔正弦余弦演算〕前記演算ブロック２２においては、前記内積演算用のハードウェアに係数テーブルや乗算器のフィードバック回路を追加することにより、三角関数や平方根の近似値を区間分割と高次の多項式展開で求められるようにすることができる。例えば前記変換行列は回転変換のとき正弦及び余弦を含むことになる。必要な角度の全てについて正弦及び余弦のデータテーブルを持つ場合には、それによるチップ面積の増大を無視することはできない。

【００５４】ここでは、前記演算ブロック２２を利用して正弦と余弦の近似値を求めることについて説明する。以下に説明する構成を付加したＦＰＵは1個の命令で正弦と余弦を並行して演算する正弦余弦命令を実行する。この正弦余弦命令の仕様は図２５に示される通り、浮動小数点レジスタＦＲ[0]にロードされた角度データに対する正弦の値を演算して結果をレジスタＦＲ[n]にライトし、同じく、レジスタＦＲ[0]にロードされた角度データに対する余弦の値を演算して結果をレジスタＦＲ[n+1]にライトする。

【００５５】図２６には前記角度データのフォーマットが示される。前記角度データは、一つの浮動小数点レジスタの上位16ビットと下位16ビットの境を固定小数点位置とする32ビット固定小数点数によって回転数を表すものとされる。小数点位置を境に上位16ビットは回転数（整数）を与え、小数点位置を境に下位16ビットは１回転を２の１６乗分割して定義する。特に、下位１６ビットの内の上位２ビットは小数点以下１６ビットのデータによって特定される角度が属する象限を意味する。このような角度フォーマットにおいて、例えば３６０°は１．０であり、１６進数のビットパターンは“０００１０００”とされる。

【００５６】正弦余弦命令は、上記角度フォーマットの角度データに対して、その正弦及び余弦を、テーラ展開による多項近似に従って取得する。図２８にはその演算手法が示される。

【００５７】前記多項近似は、前記角度データの小数点以下１６ビットを中心値ｘとこの中心値に対する差分値ｄｘに分けて前記角度データに応ずる正弦と余弦の値を演算するものであり、前記多項近似に必要とされる前記中心値ｘに対する正弦又は余弦の値だけはテーブルとして保有する。前記中心値ｘは、レジスタＦＲ[0]の小数点以下7ビットの最下位を０捨１入した値とする。中心値と角度（ラジアン）との関係は図２７に例示されている。差分ｄｘはレジスタＦＲ[0]の最下位から１０ビットを符号拡張した値とする。多項近似ではテーラ展開を用いるため、角度をラジアンで表現するように、各項の係数が与えられている。図２８に示されるＳ１～Ｓ１２は、乗算器２２０a（ＦＭ0），乗算器２２０ｂ（ＦＭ１），乗算器２２０ｃ（ＦＭ2），乗算器２２０ｄ（ＦＭ３），4入力加算器２２６を用いた演算処理の内容を式で示している。Ｓ９において多項近似式による正弦の近似値が求められ（図２８にはその多項近似式が示されている）、Ｓ１２において多項近似式による余弦の近似値が求められる（図２８にはその多項近似式が示されている）。

【００５８】前記Ｓ１～Ｓ１２の演算は（１）～（４）で示されるパイプラインで処理される。前述のように、

角度データの下位１６ビットの内の上位２ビットは小数点以下１６ビットのデータによって特定される角度が属する象限を意味する。したがって、制御部は、前記上位２ビットのデコード結果に従って（その角度データによって特定される角度が属する象限にしたがって）、前記多項近似によるＳ１０、Ｓ１２の演算結果の符号反転とその演算結果をレジスタＦＲ[n]又はＦＲ[n+1]のどちらに格納するかの選択を制御して、前記象限に応ずる正弦及び余弦の値を夫々に割り当てられたレジスタＦＲ[n]又はＦＲ[n+1]に格納することになる。象限毎の上記反転動作とレジスタ選択動作は図２８に示される通りである。

【００５９】図２９には前記正弦余弦命令を実行するための係数テーブルと乗算器のフィードバック系を付加した前記乗算器近傍のブロックダイヤグラムが示される。図２９の回路の基本は図4の演算ブロックであり、図4の演算ブロック２２に対して、8ビット及び6ビット符号拡張器３００、係数テーブル３０１、セレクタ３０２～３１２が追加された点が異なるだけである。レジスタＦＲ[0]の角度データはポートＢ[０]から与えられる。8ビット及び6ビット符号拡張器３００はレジスタＦＲ[0]の最下位10ビットから差分ｄｘを生成する回路である。係数テーブル３０１は図２７に示す中心値に応ずる正弦又は余弦の何れか一方のデータを保有し、角度データの下位１６ビットの内の下位５ビットによって指定される角度の正弦及び余弦のデータを出力する。角度データの下位１６ビットの上位２ビットは制御部に供給される。制御部は、その２ビットの値に従って、前記４入力加算器による加算出力の選択的な反転と、加算結果を格納するレジスタＦＲ[n]又はＦＲ[n+1]の選択を制御することになる。尚、係数テーブル３０１が正弦データを持つ場合に、余弦は中心値ｘの角度をπ／２から減算した角度でテーブルを参照すればよい。係数テーブルに正弦及び余弦の双方のデータを持ってもよい。

【００６０】図３０は図２８の（１）で示される第１ステップにおけるデータの流れを太い実線によって示す。図３１は図２８の（２）で示される第２ステップにおけるデータの流れを太い実線によって示す。この図において、乗算器（ＦＭＯ）２２０ａの乗算結果がセレクタ３０８，３１１にフィードバックされているが、フィードバックは小数点以下のみとされ（上位は０とする）、＋１の効果も得るようにしている。図３２は図２８の（３）で示される第２ステップにおけるデータの流れを太い実線によって示す。図３３は図２８の（４）で示される第２ステップにおけるデータの流れを太い実線によって示す。図３２及び図３３において演算結果を２度利用する場合には乗算器の入力ラッチの更新を抑止してその値を保持する。図３０乃至図３２に示される演算制御は、正弦余弦命令をデコードする制御部が行う。

【００６１】〔データプロセッサ１の優位性〕上記ＦＰＵ２の演算ブロック２２は前述のように、内積演算命令やベクトル変換演算命令等の１個の命令を実行するとき、4個の乗算器２２０ａ～２２０ｄに８個の浮動小数点数を与えて並列動作させ、それによる乗算結果を１個の４入力加算器２２６で加算して、演算結果を得る。４入力加算器２２６の出力に対する正規化、正数化及び丸めは１個の回路２２７によって行う。図３４にはその演算処理におけるデータの流れを理解し易いように演算ブロック２２の概略を示してある。

【００６２】図３５には、上記演算ブロック２２に対する比較例が示されている。これは、一対の浮動小数点数に対する積和演算器と、その結果に対する正規化、正数化及び丸めのための回路とを２組設けて構成される。図３６には上記演算ブロック２２に対する別の比較例として、上記積和演算器と正規化、正数化及び丸めのための回路とを４組み設けたものが示されている。何れの比較例も、積和演算器と正規化、正数化及び丸めのための回路とを複数組み並列化したに過ぎない。したがって、４×４の一つの内積を演算する場合には、積和演算、正規化、正数化及び丸めのための複数個の回路を単に並列動作させるだけでは済まない。夫々の演算結果に対する相関を考慮した制御が別に必要とされる。通常は、一組の積和演算器と正規化、正数化及び丸めのための回路とを４回繰返し動作させて内積を求めることになるであろう。ベクトル変換演算の場合には更に多くの演算サイクルが必要になる。積和演算、正規化、正数化及び丸めのための複数組の回路は、パイプラインのような命令実行手法によって、対象の異なる内積演算やその他の浮動小数点命令のために並列動作されるであろう。この意味において、図３５及び図３６に示される回路構成は、種々の浮動小数点命令の演算処理能力を平均的に向上させ得るという点に特徴がある。

【００６３】上記演算ブロック２２を用いる場合には、内積演算やベクトル変換演算のための実質的な演算サイクル数を少なくすることができる。すなわち、内積演算やベクトル変換演算の高速化を実現出来る。このように、演算ブロック２２の構成は、内積演算やベクトル変換演算の高速化に特化している。また、1回の並列的な乗算及び加算によって内積を求めることができるから、２入力に対する積和演算毎に丸めを行うような処理も必要ない。これにより、内積演算のレイテンシーが短く、演算精度も高く、また、2入力に対する積和演算毎を繰り返す場合のように演算順序が異なると演算結果も相違するという事態も生じない。

【００６４】また、正規化、正数化及び丸め回路は積和演算回路と同等の回路規模を有することになるので、図３５及び図３６のように積和演算、正規化、正数化及び丸めのための回路を複数組単に並列配置した構成では、並列化によって達成しようとする平均的な演算能力の向上に比べて、並列化によるチップ面積の増大が極めて大

きくなる。この意味において、積和演算、正規化、正数化及び丸めのための回路の並列数は２が妥当と考えられている。整数演算だけなら、図３７のように積和演算器を4個並列させることも現実的であるが、整数演算の場合にはデータの桁数が限られるために浮動小数点演算に比べて演算精度は低くなってしまう。図３４の演算ブロック２２は、正規化、正数化及び丸めのための回路２２７を１個備えればよい。したがって、データプロセッサは、回路規模の増大を極力抑えて、浮動小数点で内積演算やベクトル変換演算の高速化を実現出来る。

【００６５】データプロセッサ１の浮動小数点命令は16ビット固定長であり、それ故に、浮動小数点命令におけるアドレス指定フィールドは限られ、浮動小数点レジスタは１６本とされる。このような制約の下において、浮動小数点レジスタをバックバンクとフロントバンクを持つレジスタバンク構成のレジスタファイル２１によって構成している。このとき、上記変換行列全体を格納するのに16個のレジスタを消費するため、ベクトル変換演算命令においてはフロントバンクとバックバンクの双方を利用する命令仕様とされている。前述したように、変換行列をバックバンクに配置し、ベクトルデータをフロントバンクに配置する。これにより、浮動小数点命令のビット数とレジスタ本数というリソースの制約下においても、ベクトル変換演算命令の高速実行を保証している。

【００６６】また、ベクトル変換演算命令の実行において、前述のように、一連の４回の内積演算処理は、先頭のパイプライン（1）におけるレジスタライトＲＷが、最後のパイプライン（4）におけるレジスタリードＲＲの後にされているように、パイプライン化されている。したがって、ソースレジスタとディスティネーションレジスタが同一レジスタであっても、演算対象とされるベクトルデータは不所望に失われない。これにより、多数のベクトル若しくは点に対してベクトル変換が次々に行われるとき、変換前後のデータが同一レジスタに配置されれば、16本のフロントバックを構成する浮動小数点レジスタに4個のベクトルデータをロードすれば、ベクトル変換命令を4回連続的に実行することができ、そのような16本の浮動小数点レジスタに対する演算対象データのロード又は演算結果データのメモリへのストア動作の回数が少なくて済む。この意味において、ベクトル変換演算におけるベクトルデータのソースレジスタとディスティネーションレジスタを同一にできることは、レジスタ本数が限られた中で、ベクトル変換演算を高速化するのに有用である。

【００６７】また、前記変換行列は回転変換のとき正弦及び余弦を含むことになる。必要な角度の全てについて正弦及び余弦のデータテーブル３０１を持つ場合には、それによるチップ面積の増大を無視することはできない。このとき、前記演算ブロック２２は4個の乗算器を含んでいるので、それに係数テーブル３０１や乗算器のフィードバック回路を追加することにより、三角関数や平方根の近似値を区間分割と高次の多項式展開で求められるようにすることができる。これによって、正弦及び余弦をチップ面積を増大させることなく得ることができる。特に正弦と余弦の多項近似の展開式には類似性があるので、これを利用して正弦及び余弦の値を同時（並列的）に演算するので、個別に求める場合に比べて正弦及び余弦の値を高速に得ることができる。

【００６８】図３８にはそのようなデータプロセッサを適用したデータ処理システムのブロックダイヤグラムが示される。

【００６９】同図において１は上記データプロセッサ、４０１はダイナミック・ランダム・アクセス・メモリ（ＤＲＡＭ）、４０２はＤＲＡＭ４０１に対するアドレスマルチプレクス制御やリフレッシュ制御を行うＤＲＡＭ制御部、４０３はＳＲＡＭである。ＳＲＡＭ４０３はデータプロセッサ１の作業領域やデータの一時記憶領域などに利用される。４０４はデータプロセッサ１のＯＳ（Operating System）などを保有するＲＯＭである。４０５は周辺装置制御部であり、代表的に示された外部記憶装置４０６及びキーボード４０７が接続されている。４０８はフレームバッファ４０９や図示しない描画及び表示制御論理回路を備えた表示コントローラであり、ディスプレイ４１０に対する描画制御と表示制御を行う。４１１は電源回路、４１２は代表的に示されたバスである。データプロセッサ１は3次元グラフィック処理に多用される内積演算やベクトル変換演算等を浮動小数点で高速に実行することができる。しかも、浮動小数点命令のビット数及びレジスタ本数等の限られたリソースの下で上記効果を得ることができるから、データプロセッサ１のコストも低く抑えられている。したがって、図３８のデータ処理システムは、システムのコストを抑えて、3次元グラフィック処理を高精度に且つ高速に行うことができる。したがって、コストの制約は厳しいけれども、高機能及び高速化の要請も無視出来ないような、ゲーム機や携帯情報端末などに適用して優れたデータ処理システムを実現出来る。

【００７０】以上本発明者によってなされた発明を実施形態に基づいて具体的に説明したが、本発明はそれに限定されるものではなく、その要旨を逸脱しない範囲において種々変更可能であることは言うまでもない。

【００７１】例えば、図１では説明していないが、データプロセッサはメモリマネージメントユニットなどのその他の機能ブロックを含むことができる。また、データプロセッサは、スーパースカラーアーキテクチャを採用することができる。例えば、2本のパイプを有する場合、一方のパイプではベクトル変換演算命令などを実行し、他方のパイプではベクトル変換演算命令のためのベクトルデータをメモリからレジスタファイルのロードしたり、ベクトル変換演算の結果をレジスタファイルから

メモリにストアすることができる。

【００７２】また、乗算器の並列配置個数は4個以上であってもよい。また、指数処理部や符号処理部の構成、4入力加算器の構成は上記実施例に限定されず適宜変更可能である。

【００７３】また、本発明のデータプロセッサはゲーム機や携帯情報端末の制御に適用される場合に限定されず、種々の機器組み込み制御などの用途に広く利用することができる。

【００７４】

【発明の効果】本願において開示される発明のうち代表的なものによって得られる効果を簡単に説明すれば下記の通りである。

【００７５】すなわち、データプロセッサは、浮動小数点による内積演算やベクトル変換演算の高速化を実現できる。

【００７６】データプロセッサは、正規化等のための回路を１個備えればよいから、回路規模の増大を極力抑えて、浮動小数点で内積演算やベクトル変換演算の高速化を実現できる。

【００７７】ベクトル変換演算命令においてはフロントバンクとバックバンクの双方を利用する命令仕様とされており、変換行列をバックバンクに配置し、ベクトルデータをフロントバンクに配置することにより、浮動小数点命令のビット数とレジスタ本数というリソースの制約下においても、ベクトル変換演算命令の高速実行を保証できる。

【００７８】また、ベクトル変換演算命令の実行において、一連の４回の内積演算処理は、先頭の内積演算処理におけるレジスタライトが、最後の内積処理におけるレジスタリードの後にされるように、各内積処理のレイテンシーを制御するから、ソースレジスタとディスティネーションレジスタが同一レジスタであっても、演算対象とされるベクトルデータは不所望に失われない。これにより、浮動小数点レジスタに対する演算対象データのロード又は演算結果データのメモリへのストア動作の回数が少なくて済み、レジスタ本数が限られた中で、ベクトル変換演算を高速化するのに有用である。

【００７９】また、前記演算部に係数テーブルや乗算器のフィードバック回路を追加することにより、三角関数の近似値を区間分割と高次の多項式展開で求められるようにすることにより、変換行列等に利用される正弦及び余弦を、チップ面積を大幅に増大させることなく得ることができる。

【００８０】データプロセッサは、浮動小数点命令のビット数及びレジスタ本数等の限られたリソースの下において、3次元グラフィック処理に多用される内積演算やベクトル変換演算等を浮動小数点で高速に実行することができるから、前記データプロセッサを適用したデータ処理しステムは、システムのコストを抑えて、3次元グラフィック処理を高精度に且つ高速に行うことができる。したがって、コストの制約は厳しいけれども、高機能及び高速化の要請も無視出来ないような、ゲーム機や携帯情報端末などに適用して優れたデータ処理システムを実現出来る。

【図面の簡単な説明】

【図１】本発明の一例に係るデータプロセッサのブロック図である。

【図２】図1のデータプロセッサの主なレジスタ構成の説明図である。

【図３】前記ＦＰＵの一例ブロック図である。

【図４】演算ブロックの一例ブロック図である。

【図５】最大指数選択部の一例ブロック図である。

【図６】符合選択部の一例ブロック図である。

【図７】アライナの一例ブロック図である。

【図８】４入力加算器の一例ブロック図である。

【図９】４入力加算器においてＩｎｖ［ｎ］の値に対してＣｉｎ［0］，Ｃｉｎ［1］，Ｃｉｎ［2］の採り得る値を示す説明図である。

【図１０】キャリー保存加算器アレイとキャリー伝達加算器アレイの詳細な一例ブロック図である。

【図１１】４－２コンプレッサの一例説明図である。

【図１２】レジスタファイルの一例ブロック図である。

【図１３】レジスタファイルを構成する各レジスタグループの構成説明図である。

【図１４】レジスタグループの各レジスタ回路の一例構成図である。

【図１５】レジスタファイルのリードポートＡの動作態様説明図である。

【図１６】レジスタファイルのリードポートＢの動作態様説明図である。

【図１７】レジスタファイルのリードポートＣの動作態様説明図である。

【図１８】内積の応用例を示す説明図である。

【図１９】内積演算命令による処理の概略を示すブロック図である。

【図２０】内積演算命令におけレジスタファイルの利用に関する仕様の一例説明図である。

【図２１】ベクトル変換演算の一般的に示す説明図である。

【図２２】ベクトル変換演算命令による処理の概略を示すブロック図である。

【図２３】ベクトル変換演算命令におけレジスタファイルの利用に関する仕様の一例説明図である。

【図２４】一つのベクトル変換命令による複数回の内積演算処理のパイプラインを示す説明図である。

【図２５】正弦余弦命令の仕様説明図である。

【図２６】正弦余弦命令に利用される角度データのフォーマット説明図である。

【図２７】多項近似のための中心値と角度（ラジアン）

との関係を示す説明図である。

【図２８】多項近似に従った正弦余弦命令による演算処理の流れ図である。

【図２９】正弦余弦命令を実行するための係数テーブルと乗算器のフィードバック系を付加した前記乗算器近傍のブロック図である。

【図３０】図２８の（１）で示される第１ステップにおけるデータの流れを太い実線によって示す説明図である。

【図３１】図２８の（２）で示される第２ステップにおけるデータの流れを太い実線によって示す説明図である。

【図３２】図２８の（３）で示される第２ステップにおけるデータの流れを太い実線によって示す説明図である。

【図３３】図２８の（４）で示される第２ステップにおけるデータの流れを太い実線によって示す説明図である。

【図３４】内積演算命令やベクトル変換演算命令を実行するときの演算処理におけるデータの流れを理解し易いように演算ブロック２２を概略的に示したブロック図である。

【図３５】積和演算器と正規化、正数化及び丸めのための回路とを２組み単に並列配置したものを示す比較説明図である。

【図３６】積和演算器と正規化、正数化及び丸めのための回路とを４組み単に並列配置したものを示す比較説明図である。

【図３７】整数演算のために積和演算器を4個並列させたものを示す比較説明図である。

【図３８】データプロセッサを適用したデータ処理システムの一例ブロック図である。

【符号の説明】

１　データプロセッサ

２　浮動小数点ユニット

３　ＣＰＵ

４　データバス

５　命令バス

２０　転送ブロック

２１　レジスタファイル

２２　演算ブロック

２３　制御部

A[0]，A[1]，A[2]，A[3]　リードポート

B[0]，B[1]，B[2]，B[3]　リードポート

Ｃ　リードポート

Ｘ，Ｙ　ライトポート

ｆ０～ｆ１５　フロントバンクを構成するレジスタ

ｂ０～ｂ１５　バックバンクを構成するレジスタ

２２０ａ～２２０ｄ　乗算器

２２１ａ～２２１ｄ　アライナ

２２２ａ～２２２ｄ　加算器

２２３　最大指数選択部

２２４ａ～２２４ｄ　減算機

２２８　指数処理部

ＥＯＲ１～ＥＯＲ８　排他的論理ゲート

２２５　符合選択部

２２９　符号処理部

２２６　４入力加算器

２２７　正規化、正数化及び丸め回路

Ｌａ０～Ｌａ３，Ｌｂ０～Ｌｂ３　信号線群

Ｅｍａｘ　正規化前の指数

Ｓｍａｘ　正規化前の符号

Ｉｎｖ[0]，Ｉｎｖ[1]，Ｉｎｖ[2]，Ｉｎｖ[3]　乗算結果に対する符合

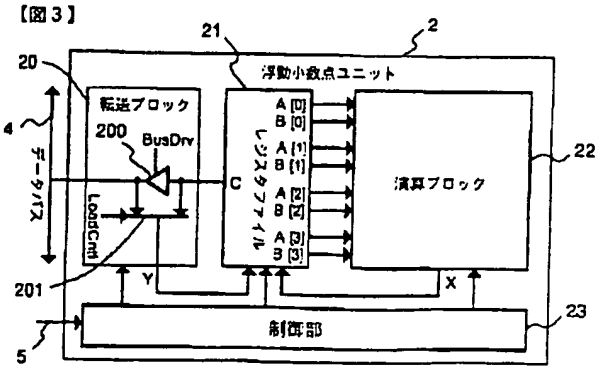
３００　８ビット及び６ビット符号拡張器

３０１　正弦余弦テーブル

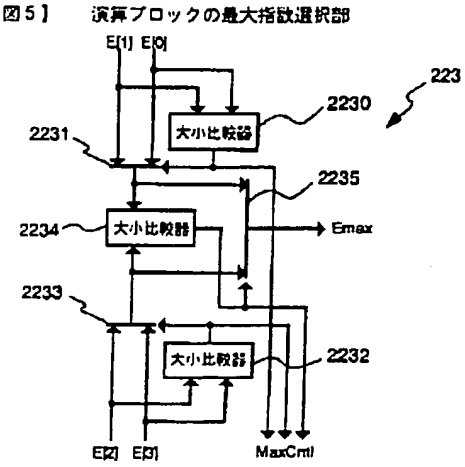
４０３　ＳＲＡＭ

４１２　バス

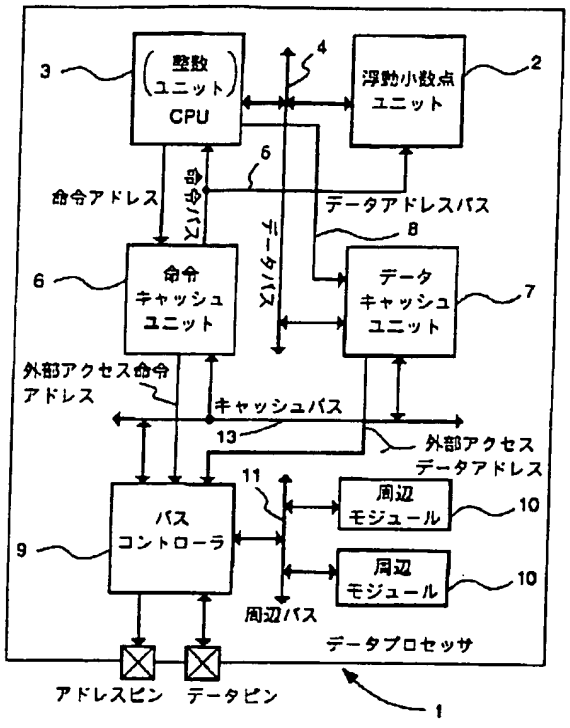
【図３】



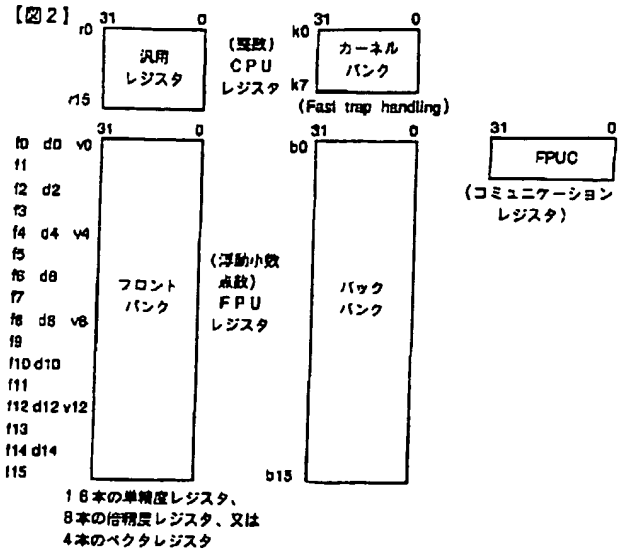
【図５】



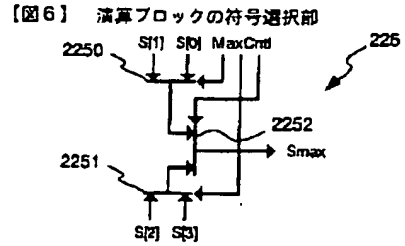
【図１】



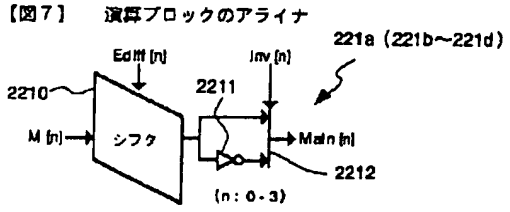
【図２】



【図６】



【図７】

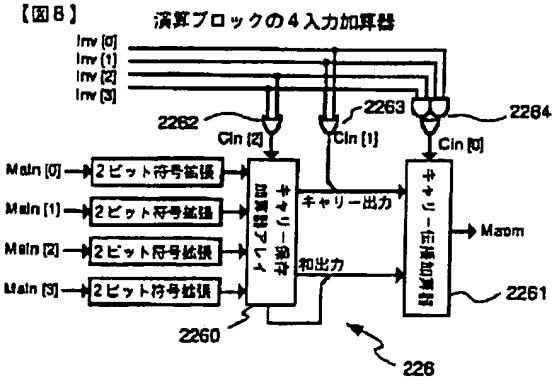


【図９】

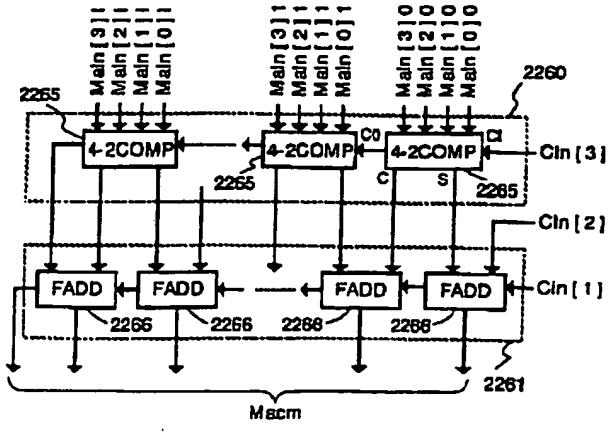
【図９】 キャリー入力変換論理

Inv				Cin		
[0]	[1]	[2]	[3]	[0]	[1]	[2]
0	0	0	0	0	0	0
0	0	0	1	0	0	1
0	0	1	0	0	0	1
0	1	0	0	0	1	0
1	0	0	0	0	1	0
0	0	1	1	1	0	1
0	1	0	1	0	1	1
0	1	1	0	0	1	1
1	0	0	1	0	1	1
1	0	1	0	0	1	1
1	1	0	0	1	1	0
0	1	1	1	1	1	1
1	0	1	1	1	1	1
1	1	0	1	1	1	1
1	1	1	0	1	1	1
1	1	1	1	1	1	1
1	1	1	1	使らない		

【図８】

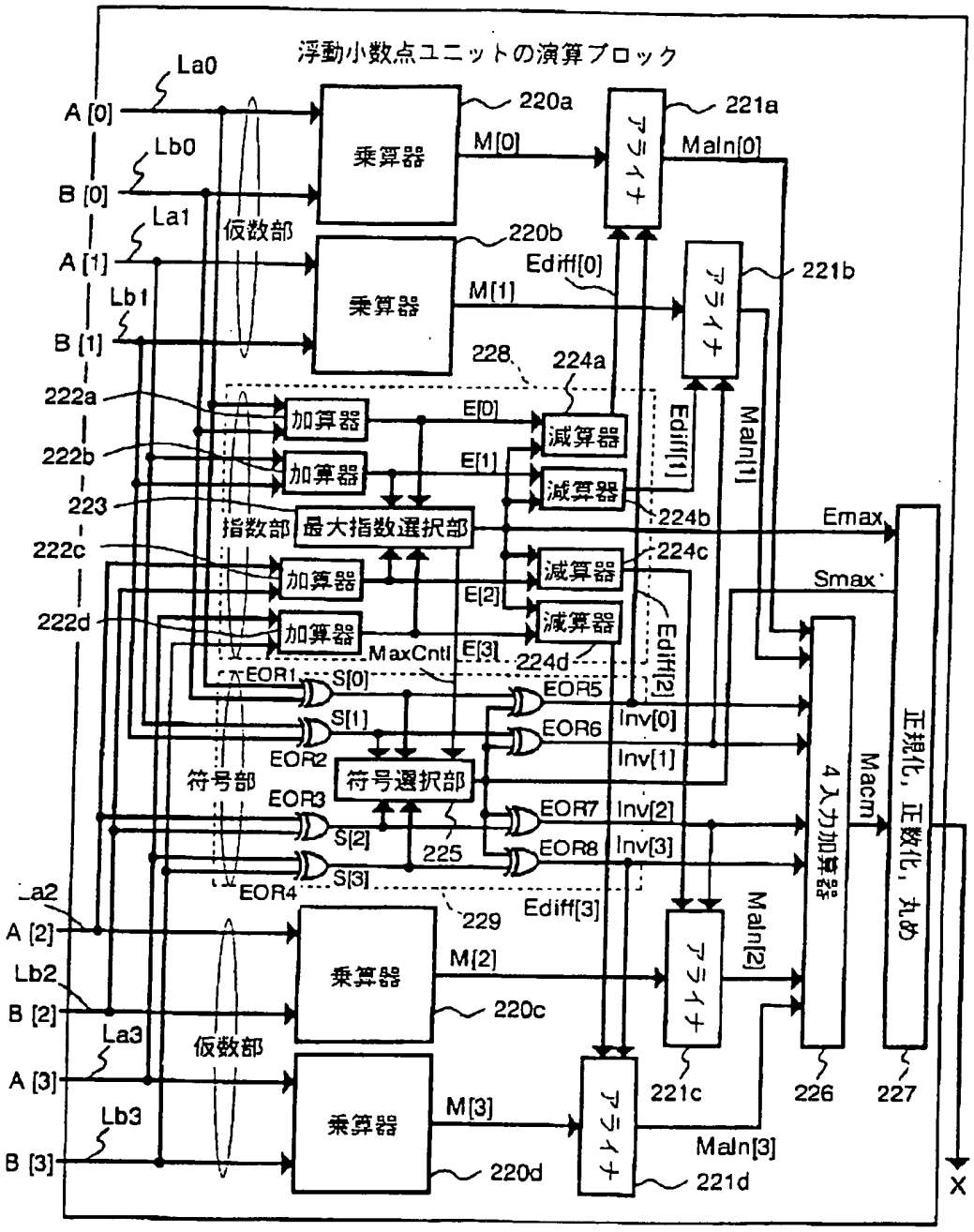


【図１０】



【図４】

【図４】



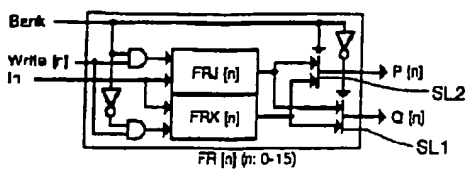
【図１４】

【図２０】

【図２５】

【図１４】

レジスタグループの各レジスタ回路の構成図



【図２０】

内積命令仕様

$$FR[n+3] = (V[m], V[n]) \quad \text{但し，} V[n] = \begin{pmatrix} FR[n] \\ FR[n+1] \\ FR[n+2] \\ FR[n+3] \end{pmatrix} \quad (n : 0, 4, 8, 12)$$

$$FR[n] = \begin{cases} FRJ[n] & Bank = 0\text{の場合} \\ FRK[n] & Bank = 1\text{の場合} \end{cases}$$

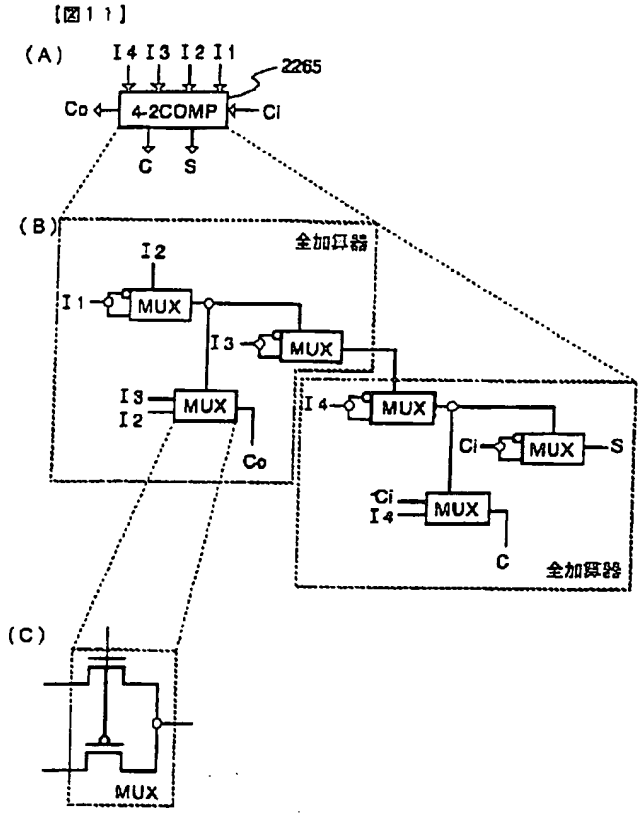
【図２５】

正弦余弦命令仕様

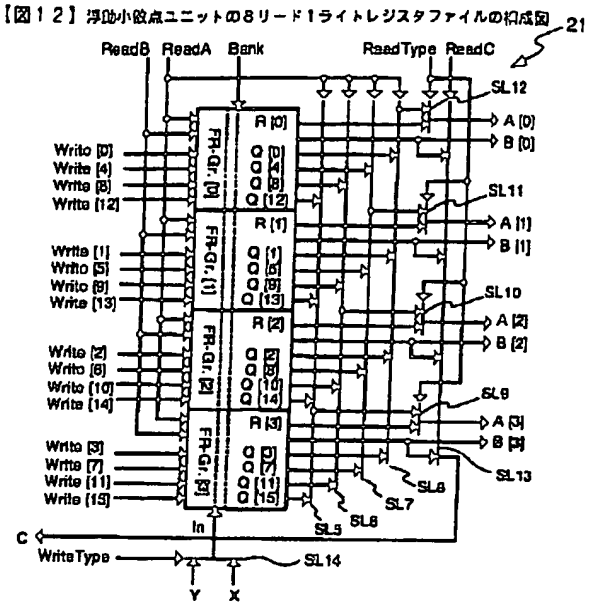
FR [n] = sin FR [0]

FR [n+1] = cos FR [0]

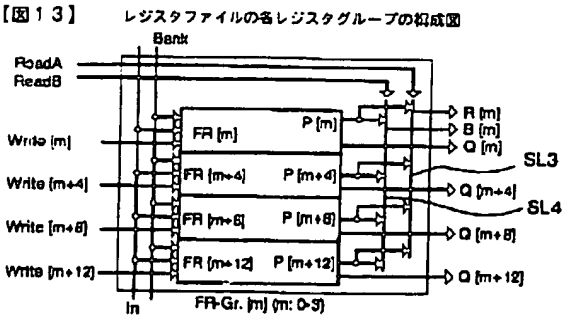
【図１１】



【図１２】



【図１３】



【図１５】

【図１５】 レジスタファイルのリード動作A

RecdType	Bank	RecdA	A [0]	A [1]	A [2]	A [3]
0	0	0	FRJ [0]	FRJ [1]	FRJ [2]	FRJ [3]
		1	FRJ [4]	FRJ [5]	FRJ [6]	FRJ [7]
		2	FRJ [8]	FRJ [9]	FRJ [10]	FRJ [11]
		3	FRJ [12]	FRJ [13]	FRJ [14]	FRJ [15]
	1	0	FRK [0]	FRK [1]	FRK [2]	FRK [3]
		1	FRK [4]	FRK [5]	FRK [6]	FRK [7]
		2	FRK [8]	FRK [9]	FRK [10]	FRK [11]
		3	FRK [12]	FRK [13]	FRK [14]	FRK [15]
1	0	0	FRK [0]	FRK [4]	FRK [8]	FRK [12]
		1	FRK [1]	FRK [5]	FRK [9]	FRK [13]
		2	FRK [2]	FRK [6]	FRK [10]	FRK [14]
		3	FRK [3]	FRK [7]	FRK [11]	FRK [15]
	1	0	FRJ [0]	FRJ [4]	FRJ [8]	FRJ [12]
		1	FRJ [1]	FRJ [5]	FRJ [9]	FRJ [13]
		2	FRJ [2]	FRJ [6]	FRJ [10]	FRJ [14]
		3	FRJ [3]	FRJ [7]	FRJ [11]	FRJ [15]

【図１６】

【図１６】

レジスタファイルのリード動作B

Bank	ReadB	B [0]	B [1]	B [2]	B [3]
0	0	FRJ [0]	FRJ [1]	FRJ [2]	FRJ [3]
	1	FRJ [4]	FRJ [5]	FRJ [6]	FRJ [7]
	2	FRJ [8]	FRJ [9]	FRJ [10]	FRJ [11]
	3	FRJ [12]	FRJ [13]	FRJ [14]	FRJ [15]
1	0	FRK [0]	FRK [1]	FRK [2]	FRK [3]
	1	FRK [4]	FRK [5]	FRK [6]	FRK [7]
	2	FRK [8]	FRK [9]	FRK [10]	FRK [11]
	3	FRK [12]	FRK [13]	FRK [14]	FRK [15]

【図２１】

【図２１】

$$A \times p\,I \rightarrow p'I$$

$$\begin{bmatrix} a11 & a12 & a13 & a14 \\ a21 & a22 & a23 & a24 \\ a31 & a32 & a33 & a34 \\ a41 & a42 & a43 & a44 \end{bmatrix} \times \begin{bmatrix} XI \\ YI \\ ZI \\ WI \end{bmatrix} \longrightarrow \begin{bmatrix} X'I \\ Y'I \\ Z'I \\ W'I \end{bmatrix}, \text{for } I = 1, n$$

単一命令 ftrv back,Vnにより単精度浮動小数点で処理

【図１７】

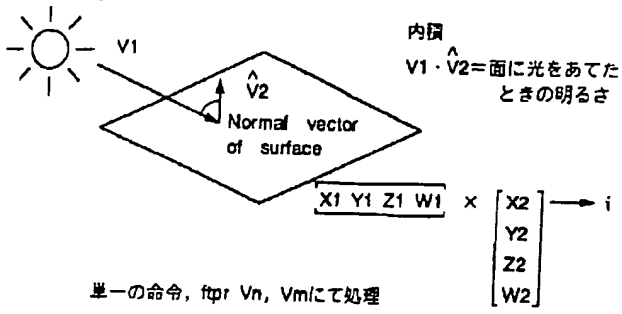
【図１８】

【図１７】

レジスタファイルのリード動作C

Bank	ReadB \ ReadC	0	1	2	3
0	0	FRJ [0]	FRJ [1]	FRJ [2]	FRJ [3]
	1	FRJ [4]	FRJ [5]	FRJ [6]	FRJ [7]
	2	FRJ [8]	FRJ [9]	FRJ [10]	FRJ [11]
	3	FRJ [12]	FRJ [13]	FRJ [14]	FRJ [15]
1	0	FRK [0]	FRK [1]	FRK [2]	FRK [3]
	1	FRK [4]	FRK [5]	FRK [6]	FRK [7]
	2	FRK [8]	FRK [9]	FRK [10]	FRK [11]
	3	FRK [12]	FRK [13]	FRK [14]	FRK [15]

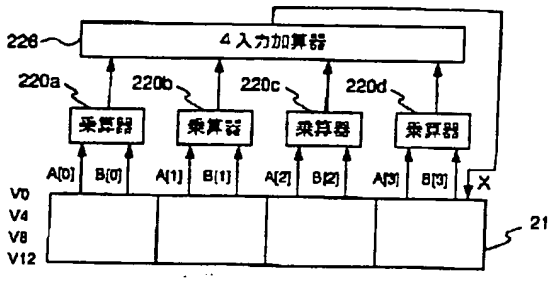
【図１８】



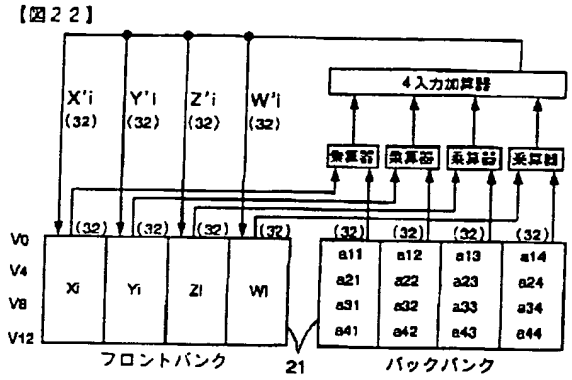
【図１９】

【図２２】

【図１９】



【図２２】



【図２３】

【図２３】　ベクトル変換命令仕様

$$V[n] = Matrix \times V[n]$$

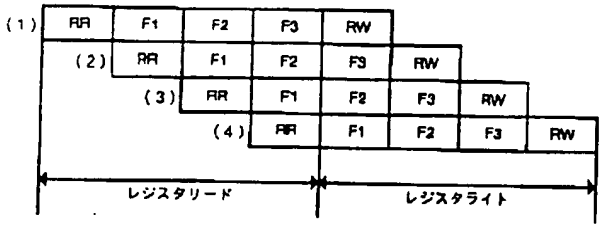
但し，$Matrix = \begin{pmatrix} FB[0] & FB[4] & FB[8] & FB[12] \\ FB[1] & FB[5] & FB[9] & FB[13] \\ FB[2] & FB[6] & FB[10] & FB[14] \\ FB[3] & FB[7] & FB[11] & FB[15] \end{pmatrix}$

$FB[n] = \begin{cases} FRK[n] & Bank = 0の場合 \\ FRJ[n] & Bank = 1の場合 \end{cases}$

【図２４】

【図２４】

ベクトル変換命令処理パイプライン



(1) $FR[n] = (FB[0]\ FB[4]\ FB[8]\ FB[12])\ V[n]$
(2) $FR[n+1] = (FB[1]\ FB[5]\ FB[9]\ FB[13])\ V[n]$
(3) $FR[n+2] = (FB[2]\ FB[6]\ FB[10]\ FB[14])\ V[n]$
(4) $FR[n+3] = (FB[3]\ FB[7]\ FB[11]\ FB[15])\ V[n]$

RR: レジスタリードステージ
F1: 第一演算ステージ
F2: 第二演算ステージ
F3: 第三演算ステージ
RW: レジスタライトステージ

【図２６】

【図２６】

正弦余弦命令の角度フォーマット

上位16ビット	下位16ビット

小数点

小数点以下16ビットの32ビット固定少数点数で回転数を表わす。

【図２７】

【図２８】

【図２７】

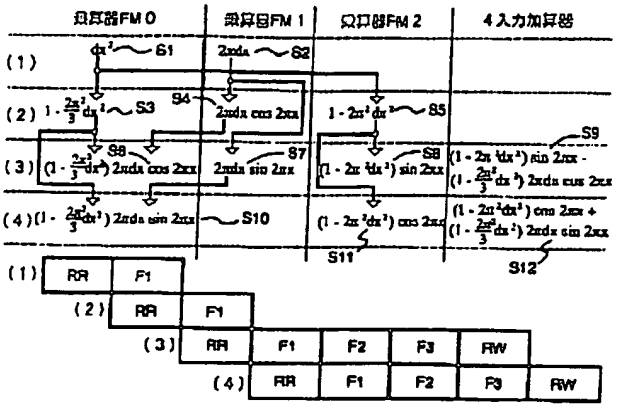
角度入力の下位14ビットの内の上位５ビットの値に対応する線依展開の中心値

角度入力	線依展開の中心値		角度入力	線依展開の中心値	
	内部表現	ラジアン		内部表現	ラジアン
11111 00000	0.000000	0	01111 10000	0.001000	π/4
00001 00010	0.000001	π/32	10001 10010	0.001001	9π/32
00011 00100	0.000010	π/16	10011 10100	0.001010	5π/16
00101 00110	0.000011	3π/32	10101 10110	0.001011	11π/32
00111 01000	0.000100	π/8	10111 11000	0.001100	3π/8
01001 01010	0.000101	5π/32	11001 11010	0.001101	13π/32
01011 01100	0.000110	3π/16	11011 11100	0.001110	7π/16
01101 01110	0.000111	7π/32	11101 11110	0.001111	15π/32

【図２８】

正弦余弦命令の処理フロー

xは FR [n] の小数点以下７ビットの以下位を０含１入した位
dxは FR [n] の以下位10ビットを符号拡張した位
以算器FM０，FM２では小数点以下のみフィードバックして上位を０とし，＋１と同じ効果を得る演算結果を２位使用する場合は以算器の入力ラッチの更新を抑止して値を保持する

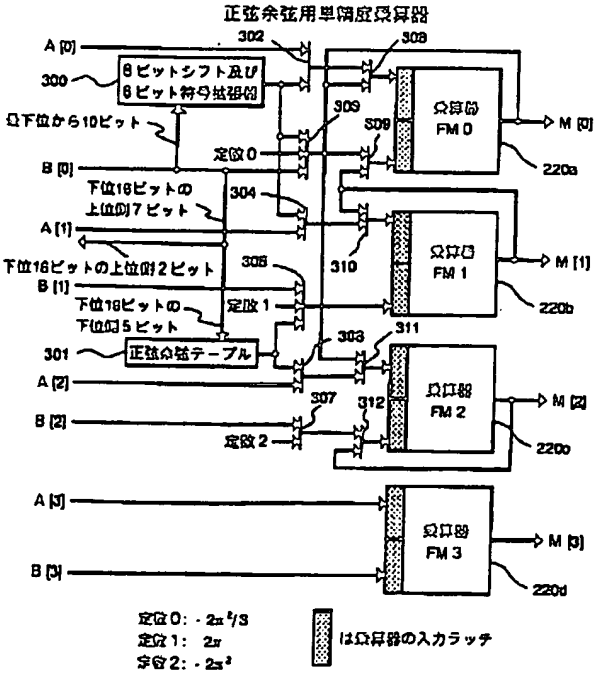


x＋dx が第Ⅰ象限:
ステップ（３）で４入力加算器出力を位処理（正規化，正位化，丸め）してFR [n] へ
ステップ（４）で４入力加算器出力を位処理してFR [n+1] へ
x＋dx が第Ⅱ象限:
ステップ（３）で４入力加算器出力を符号反転および位処理してFR [n+1] へ
ステップ（４）で４入力加算器出力を位処理してFR [n] へ
x＋dx が第Ⅲ象限:
ステップ（３）で４入力加算器出力を符号反転および位処理してFR [n] へ
ステップ（４）で４入力加算器出力を符号反転および位処理してFR [n+1] へ
x＋dx が第Ⅳ象限:
ステップ（３）で４入力加算器出力を位処理してFR [n+1] へ
ステップ（４）で４入力加算器出力を符号反転および位処理してFR [n] へ

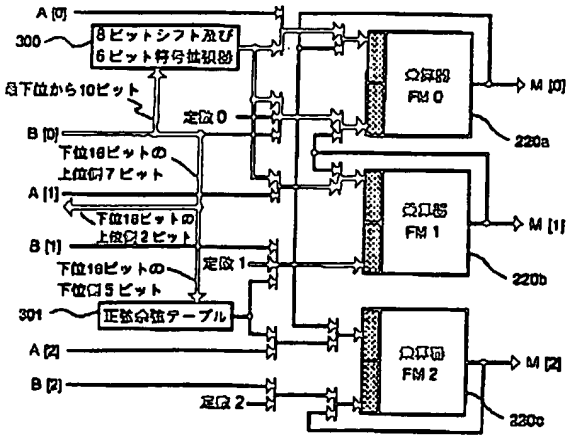
【図２９】

【図３０】

【図２９】

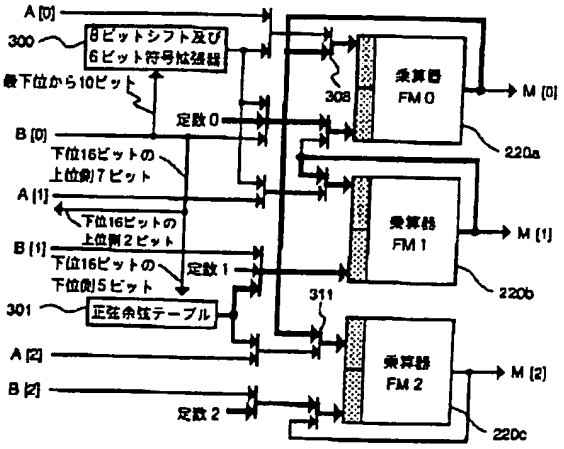


【図３０】 正弦余弦用単精度乗算器の１ステップ動作



【図３１】

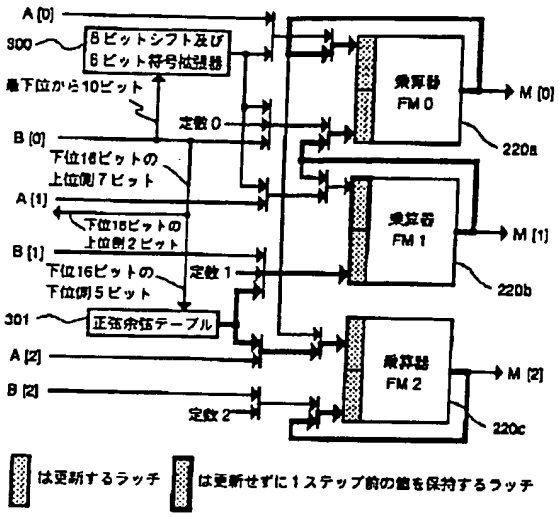
【図３１】 正弦余弦用単精度乗算器第２ステップ動作



【図３２】

【図３２】

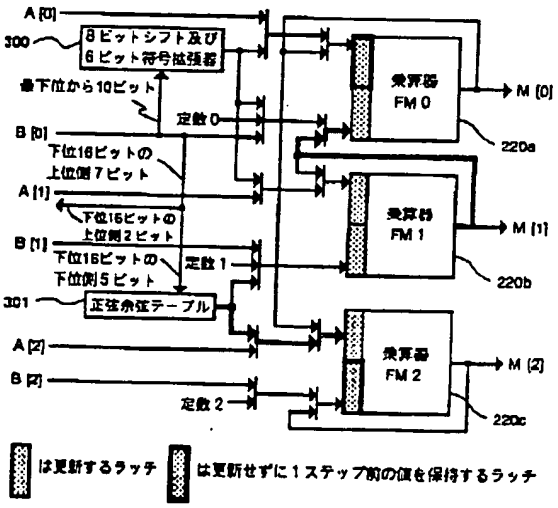
正弦余弦用単精度乗算器第３ステップ動作



【図３３】

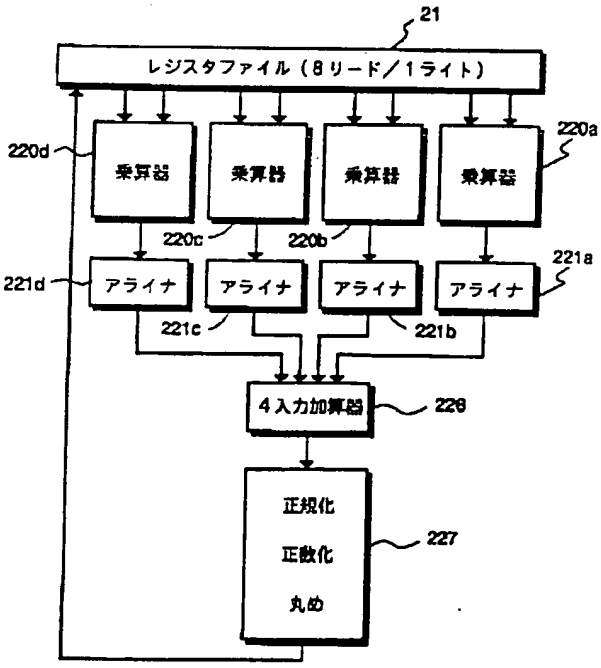
【図３３】

正弦余弦用単精度乗算器第４ステップ動作



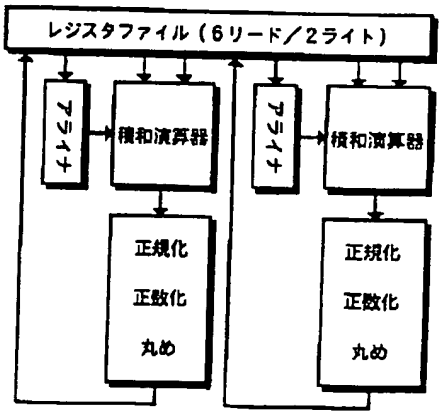
【図３４】

【図３４】



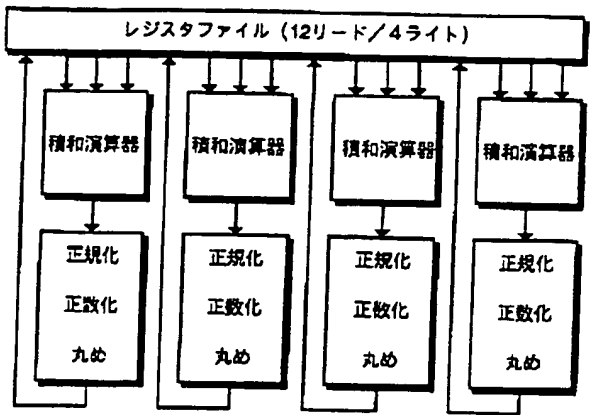
【図３５】

【図３５】 ２並列浮動小数積和演算

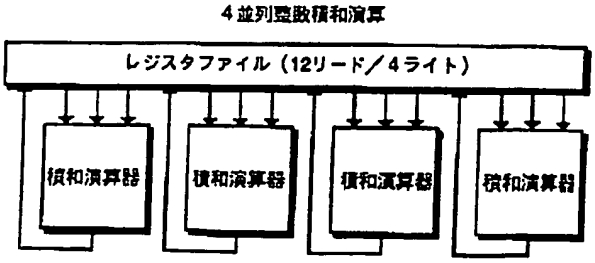


【図３６】

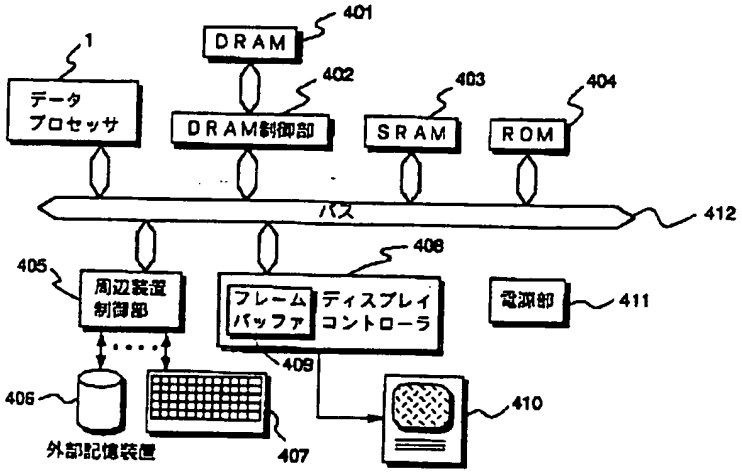
【図３６】



【図３７】



【図３８】



【図３８】

フロントページの続き

(51)Int.Cl.[6]	識別記号	F I	
// Ｇ０６Ｔ　1/00		Ｇ０６Ｆ　15/66	Ｊ

(72)発明者　戸塚　米太郎
東京都国分寺市東恋ケ窪一丁目280番地
株式会社日立製作所中央研究所内